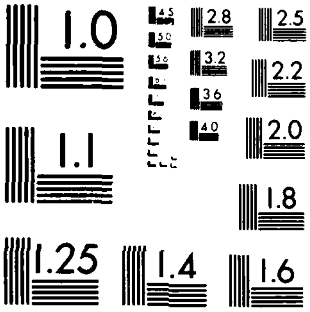
AD-A141 556 CARRIER AIR STRIKE AND ATTACK MODULE(U) KAMAN SCIENCES CORP ARLINGTON VA J A VERCOE ET AL. 24 APR 84 K-84-69U(R) N00014-82-C-0382 1/1

UNCLASSIFIED F/G 15/7 NL

END
DATE
FILMED
7 84
DTIC



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

(1)

CARRIER AIR STRIKE AND ATTACK MODULE

K-84-69U(R)

24 APRIL 1984

KAMAN SCIENCES
CORPORATION
SYSTEMS DIRECTORATE
1911 Jefferson Davis Hwy., Arlington, VA 22202

KAMAN SCIENCES CORPORATION
a Kaman Company
SYSTEMS DIRECTORATE • Telephone (703) 521-0800
1911 Jefferson Davis Hwy. • Suite 1200 • Arlington, VA 22202

CARRIER AIR STRIKE AND ATTACK MODULE

K-84-69U(R)

24 APRIL 1984

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER
K-84-69U(R)

2. GOVT ACCESSION NO.
AD-A141556

3. RECIPIENT'S CATALOG NUMBER

4. TITLE (and Subtitle)
CARRIER AIR STRIKE AND ATTACK MODULE

5. TYPE OF REPORT & PERIOD COVERED
FINAL REPORT
15 April 1982 - 29 April 1984

6. PERFORMING ORG. REPORT NUMBER

7. AUTHOR(s)
Janine A. Vercoe
Michael O. Brosee

8. CONTRACT OR GRANT NUMBER(s)
N00014-82-C-0382

9. PERFORMING ORGANIZATION NAME AND ADDRESS
Kaman Sciences Corporation
1911 Jefferson Davis Highway, Suite 1200
Arlington, Virginia 22202

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS

11. CONTROLLING OFFICE NAME AND ADDRESS
Office of Naval Research
Department of the Navy
800 N. Qunicy St., Arlington, VA 22217

12. REPORT DATE
24 April 1984

13. NUMBER OF PAGES
72

14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)
Office of the Chief of Naval Operations
Department of the Navy, OP-654D
Washington, DC 20350

15. SECURITY CLASS. (of this report)
UNCLASSIFIED

15a. DECLASSIFICATION DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

This document has been approved for public release and sale; its distribution is unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

Unlimited.

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

CASAM
NNWS
SIMULATION
INTERFACE
AIRRAID
AIR OPERATIONS
AIRCRAFT
BATTLE GROUP
CARRIER
EVENTS
LAUNCH
LAUNCH COMPLETE
RECOVERY
RECOVERY COMPLETE
MAINTENANCE
MAINTENANCE COMPLETE
STATION

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This report documents the design specification of the Carrier Air Strike and Attack Module. The module is designed to be integrated with the Naval Nuclear Warfare Simulation. ~~(NNWS).~~ The module is utilized to maintain the accountability of a carrier's aircraft by updating the carrier's Status Board. The Status Board is utilized to inform the player of the availability of aircraft that the player will make judgements based on the capabilities of the carriers.

TABLE OF CONTENTS

TABLE OF CONTENTS

1.0 INTRODUCTION

1.1 Purpose

The Chief of Naval Operations (OP-654) is concurrently developing the Battle Group (BG) entity, the Air Raid (AR) entity, and an Air Operations (AIROPS) module for use in the Naval Nuclear Warfare Simulation (NNWS). These will add to the simulation the capability to model attacks on enemy Naval units and strikes on land targets with carrier based aircraft. These additions to NNWS will allow the player to utilize the entire range of Naval assets available to a Carrier Battle Group (CVBG) commander.

The purpose of the proposed Carrier Air Strike and Attack Module (CASAM) is to simulate realistically carrier operations under both current and future environments, taking into consideration resource (fuel/aircraft ordnance) reduction and the varying combat availability of the task force air power. To accomplish this end, CASAM maintains accountability of the carrier aircraft's missions, locations and availabilities.

1.2 Scope

This System Change Proposal (SCP) recommends methods and procedures for deriving proper schedules for simulated launches and recoveries by the carriers involved, recommends procedures and methods to be used in the routines required to control the carrier's aircraft, and outlines the methods to be used to input the carrier's aircraft to the various engagement models which will be utilized with NNWS. Currently only one engagement model is operative and that is the Naval Air Defense Simulation (NADS) (Reference 1) and this model will have data transferred by the NADS Interface program.

The CASAM module is to be divided into the following sections:

Initialization
Flight Scheduling Routines
Aircraft Controlling Events
Data Relationship

Each Section is described in detail separately.

Section 3.0 will provide the capabilities to schedule an aircraft. The next section develops and executes the flight environment by means of events within the IBM 3033 system. This is transparent to the player except for the reports generated.

This report constitutes the SCPs for Task 1, Task 2 and Task 3 of contract N00014-82-C-0382. Section 3.0 fulfills item number A005 of the Contract Data Requirements List (CDRL) and Section 4.0 fulfills item number A003 of the CRDL. These tasks were combined into one SCP to provide greater clarity and understanding of the utilization of CASAM within NNWS.

2.0 INITIALIZATION

2.1 Description

INITCSM is an initialization routine used to initialize the data structures within the CASAM module. This initial routine will become a subroutine which will be incorporated into the initialization routine for NNWS.

Since INITCSM adds data to the declarations within NNWS, it is necessary that the declaration definition occur before INITCSM initializes the data elements which are required by CASAM. Therefore, proper placement of this routine within the NNWS initialization routine is essential and will be coordinated with OP-654 and the Applied Physics Laboratory.

The FLIGHT_SCHEDULE is a link list containing the non-working launch sequences which are based upon a schedule number, AC_SKED#. To initialize this link list, a sequential data file called SKEDAT is created to read in the total number of schedules, SKED#, and the possible launch sequences for each flight schedule. An example of the data contained in SKEDAT appears in Figure 1. The FLIGHT_SCHEDULE declaration appears as follows:

```
DCL SKED#            FIXED BINARY(15),
    1 FLIGHT_SCHEDULE (SKED#) CONTROLLED,
      2 SLIDETIME    FIXED BINARY(31),
      2 LIST,
        3 HEAD       POINTER,
        3 TAIL       POINTER,
        3 NUM        FIXED BINARY(15);
```

```
                     **** FLIGHT SCHEDULE ****
*************************************************************
SKED# = 22;
* * * * * *
VALID# 1
SLIDETIME
  30
#SEQUENCES = 48;
ACNAME    #ACTYPE    MISSION    ORDNANCE    RTB    LAUNCHTIME
'F14'        6          4          0        585       480
'A7'         6          7          0        585       480
'A6'         3          7          0        585       480
'KA6'        1          9          0        585       480
'E2'         1          6          0        690       480
'S3'         1          5          0        690       480
'SH3'        1          5          0        690       480
'EA6'        1         11          0        690       480
'F14'        6          4          0        690       585
'A7'         6          7          0        690       585
'A6'         3          7          0        690       585
'S3'         1          5          0        690       585
'SH3'        1          5          0        690       585
  .
  .
  .
VALID# 2
SLIDETIME
 30
#SEQUENCES = 48
ACNAME    #ACTYPE    MISSION    ORDNANCE    RTB    LAUNCHTIME
'F14'        4          4          0        585       480
'A7'         4          7          0        585       480
'A6'         2          7          0        585       480
  .
  .
  .
```

FIGURE 1

```
DCL P@FLIGHT_SEQUENCE POINTER,
    1 FLIGHT_SEQUENCE BASED(P@FLIGHT_SEQUENCE),
      2 LINK        POINTER,
      2 ACTYPE      FIXED BINARY(15),
      2 #ACTYPE     FIXED BINARY(15),
      2 MISSION     FIXED BINARY(15),
      2 ORDNANCE    FIXED BINARY(15),
      2 RTB         FIXED BINARY(15),
      2 LAUNCH_TIME FIXED BINARY(31).
```

The Carrier data structure which was previously initialized, will also require the initialization of three linked lists - the Flight Schedule, the Status Board, and the Engagement (these lists are discussed in detail in Section 5.0) and an airwing composition number, AWC. The CARRIER.FLIGHT_SCHEDULE is a link list which will contain the working launch sequences that are based upon a launch time. This link list is initialized so that its pointers are null. The CARRIER.STATUS_BOARD link list will require data from a data file. The CARRDAT data file (Figure 2) is created to contain the airwing composition for each carrier. To create the airwing composition for a carrier, (CARRIER.STATUS_BOARD), a pointer will be set to point to the AC_STATUS_RECDS declaration, which is defined as a link list declaration. The CARRIER.ENGAGEMENT declaration is a link list that contains information on engagement aircraft. The CARRIER.ENGAGEMENT link list is initialized so that its pointers point to NULL. As launches are made, aircraft that are on engagement missions will be added to the Engage link list declaration.

If there are changes required in the declarations, then the data files must be updated and the routine, INITCSM, must be called to update the link lists.

```
**** CARRIER STATUS BOARD ****
******************************


* * * * ACNAME                   #ACTYPES * * * *
        'A6'                        09
        'A7'                        18
        'F14'                       24
        'KA6'                       04
        'EA6'                       04
        'E2'                        04
        'S3'                        10
        'SH3'                       08
```

FIGURE 2

2.2 CASINIT Routine:

```
Read in the number of schedules, SKED#
Allocate FLIGHT_SCHEDULE
Do I = 1 to SKED#
  o  Read in the SLIDETIME for each schedule
  o  Read in the number of flight sequences, #SEQUENCES

Do J = 1 to #SEQUENCES
  o  Allocate FLIGHT_SEQUENCE
  o  Read in data required for the FLIGHT_SEQUENCE
  o  Assign a fixed value for the aircraft type based on the
     aircraft name, ACNAME
  o  Check for a valid assignment of a mission
  o  Check for a valid assignment of an ordnance load
end /* DO J = 1 to #SEQUENCES */
end /* DO I = 1 to SKED# */

Set pointer, P@UNIT, to the head of the carrier unit list.
Do until P@UNIT equals null
  o  Read in number of aircraft type, ACNUM
  o  Do I = 1 to ACNUM
     -  Create an AC_STATUS_RECDS entity
     -  Read in aircraft name, ACNAME, and the number of
        aircraft, AC_STATUS_RECDS.#ACTYPE for that type
     -  Assign a fixed value for the aircraft type based on
        ACNAME
  o  end /*DO I = 1 to ACNUM */
  o  GO to next carrier unit in unit list
end /* DO until P@UNIT equals null */
```

3.0. FLIGHT SCHEDULE ROUTINE

The Flight Schedule Routine (CASFLT) will derive an air operations schedule that provides for the launch and recovery of a Battle Group's air assets during the simulation. It will be initiated by a call from the BGDISP Routine that selects and implements the appropriate battle group disposition for each PIM leg (Reference 2). This routine also updates or verifies a flight schedule at the beginning of each PIM. Events that call the Flight Schedule Routine are EN01 and EN31. EN01, which controls the movement of all game units, will call the Flight Schedule Routine if there is a change in the threat. EN31 will call the routine when a battle group unit arrives at a patrol area. The BGDISP Routine uses a table lookup procedure to select a battlegroup disposition and to obtain a flight schedule number (AC_SKED#). This schedule number is selected based on the following factors:

Threat
Number of carriers in the Battle group
Defense condition (DEFCON)
Mission
State
Air wing composition (AWC) on the carrier.

Figures 3 and 4 show an example of the selection procedure.

The schedule number, AC_SKED#, is passed to the Flight Schedule routine to select the appropriate launch sequences, and a pointer to the battlegroup (to obtain a pointer to the appropriate carrier(s) contained in the battlegroup's composition list). A working schedule is then created by using the Create subroutine and the $LINK macro to link the launch sequences to

DEFCON	#CV	THREAT	MISSION **	STATE ***	AIR WING COMPOSITION	SCHEDULE
LO	1	*	J	E	1	1
LO	1	*	C	O	1	2
HI	1	A	J	O	1	14
			. . .			

Thus for DEFCON = HI
#CV = 1
THREAT = AAW
MISSION = Power
Projection
AWC = 1
STATE = ON
STATION
Schedule 14 would be chosen

* Indicates a blank field which will allow any entry

** J = power projection and C = sea control

*** E = Enroute and 0 = on Station

FIGURE 3

DEFCON	#CV	THREAT	MISSION	STATE	AWC	SCHEDULE #
LO	1				1	1
LO	2				1	2
HI	1	A	J	O	1	3
HI	1	U	J	O	1	4
HI	1	S	J	O	1	5
HI	1		C	O	1	6
HI	1			E	1	7
HI	2			E	1	8
HI	2		C	O	1	9
HI	2	A	J	O	1	10
HI	2	U	J	O	1	11
HI	2	S	J	O	1	12
WAR	1	A	J	O	1	13
WAR	1	U	J	O	1	14
WAR	1	S	J	O	1	15
WAR	1		C	O	1	16
WAR	1			E	1	17
WAR	2			E	1	18
WAR	2		C	O	1	19
WAR	2	A	J	O	1	20
WAR	2	U	J	O	1	21
WAR	2	S	J	O	1	22

A blank field indicates that the field is not considered in the selection, therefore any entry will suffice.

FIGURE 4

the CARRIER.FLIGHT_SCHEDULE declaration. This will provide the launch and recovery for a carrier's air assets. The declaration for the carrier's flight schedule appears as follows:

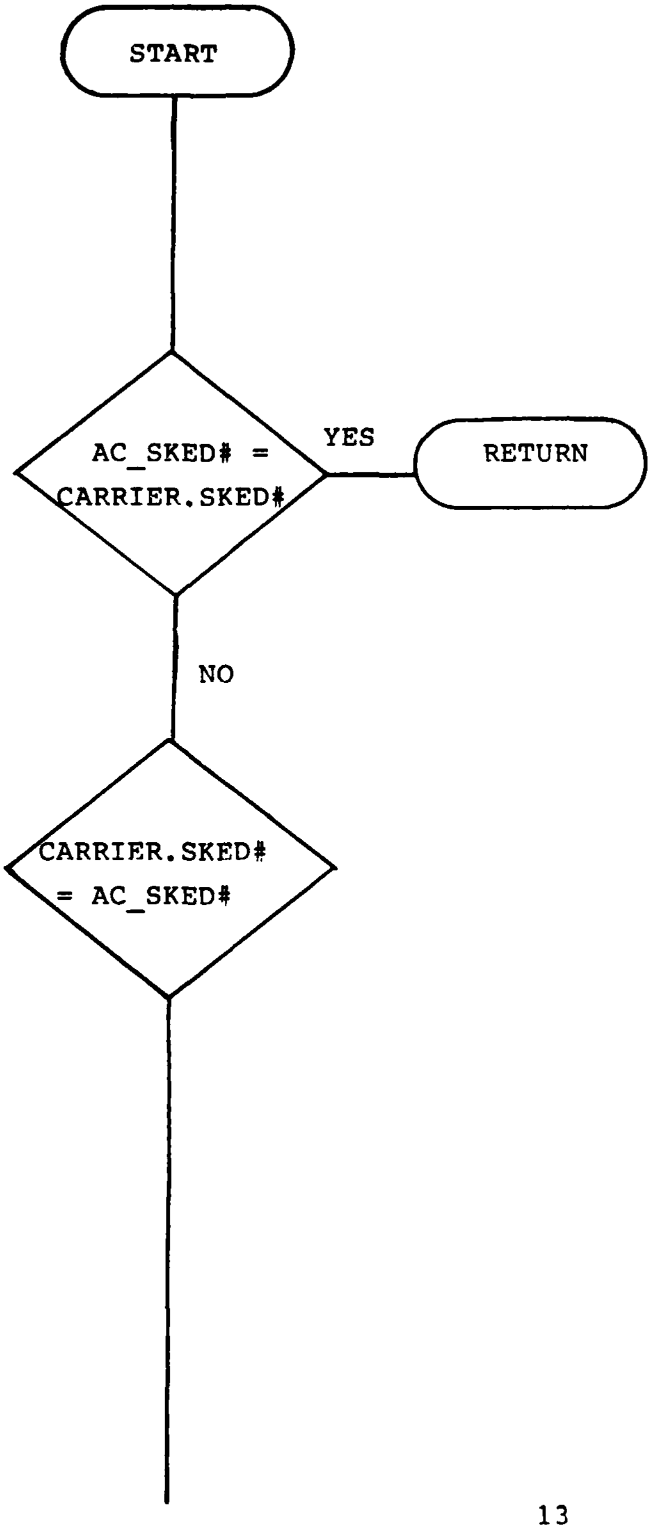
```
DCL LAUNCH_SIZE INITIAL (3/*32*/)  FIXED BINARY(15),
    P@LAUNCH                       POINTER,
    1 LAUNCH BASED (P@LAUNCH),
      2 LINK                       OFFSET(TESPACE),
      2 LAUNCH_TIME                FIXED BINARY(31),
      2 ACTYPE                     FIXED BINARY(15),
      2 #ACTYPE                    FIXED BINARY(15),
      2 MISSION                    FIXED BINARY(15),
      2 ORDNANCE                   FIXED BINARY(15),
      2 RTB                        FIXED BINARY(31).
      2 PAD                        CHARACTER(12).
```

In addition, an identification number to indicate which airwing composition is on board the carrier must be added to the Carrier declaration (see Section 5). In this first attempt of the air operations module, this air wing composition (AWC = 1) will be fixed with the composition as follows:

24	F14
18	A7
9	A6
4	KA6
4	EA6
4	E2
10	S3
8	SH3.

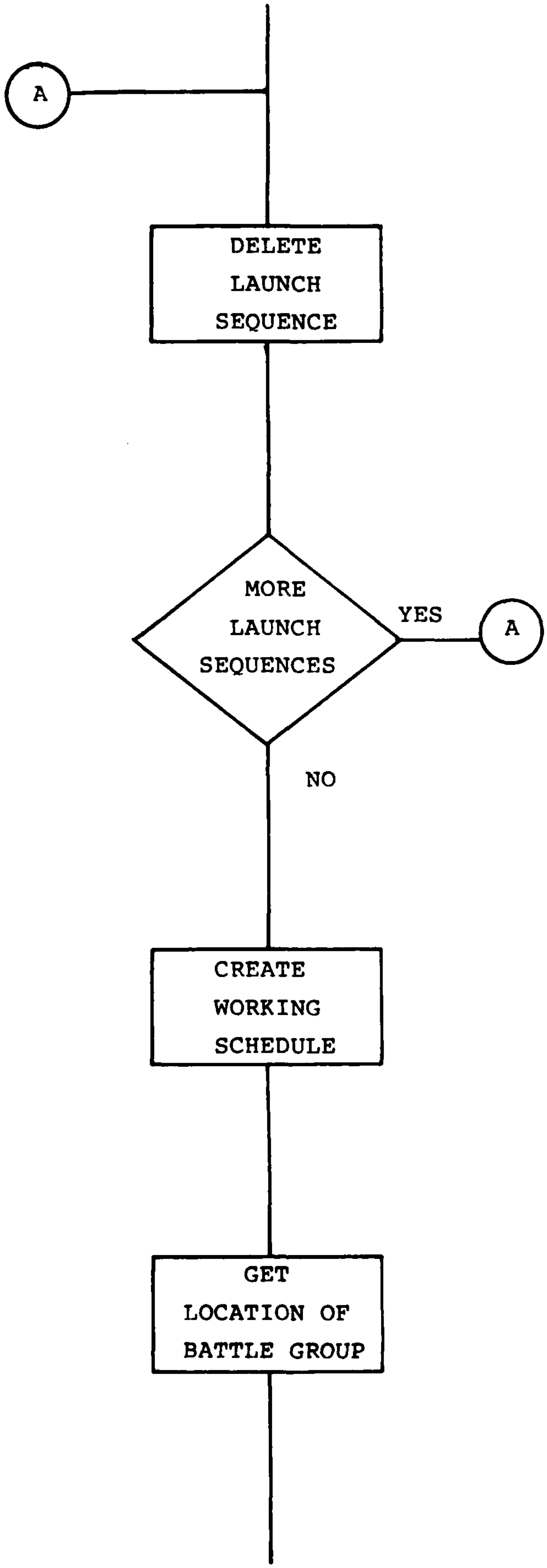
Once the flight schedule is selected and a working schedule is implemented into the CARRIER.FLIGHT_SCHEDULE link list, the launch times (local in the predetermined schedule data structure to account for day/night scheduling differences) are altered to game time based on the carrier's location. Using the $FIND macro, search for the first group of launch sequences that have a launch time greater than the current game time. The Preparation Time Routine, CASPREP, is called to evaluate if there is sufficient time between the game time and the time of the first launch event. Based on the launch time passed back from the Preparation Time Routine, a Launch Event is scheduled. This precludes the player from evaluating a threat and immediately launching sufficient aircraft to protect himself without a realistic time delay to load the proper ordnance, brief the crews and fuel the aircraft.

3.1.2 CASFLT FLOWCHART



Sequence initiated bv a call from the BGDISP Routine with a selected schedule number, AC_SKED#.

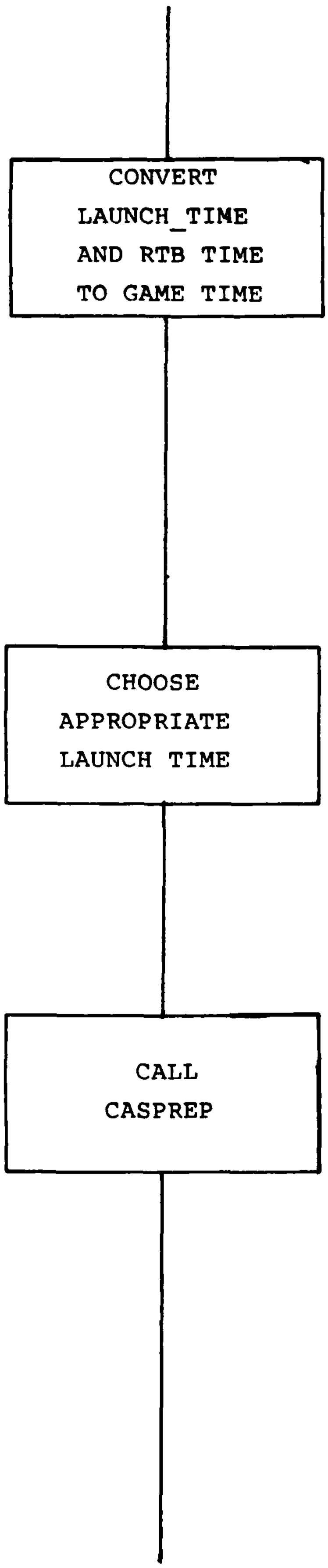
If the schedule number contained in the carrier unit equals AC_SKED#, then return to the BGDISP Routine, else assign the schedule number passed, AC_SKED#, to CARRIER.SKED#.



Cycle through the flight schedule list contained in the carrier

Delete launch sequences that currently exist on the carrier's flight schedule.

Go to the next launch sequence in the list


CONVERT LAUNCH_TIME AND RTB TIME TO GAME TIME
CHOOSE APPROPRIATE LAUNCH TIME
CALL CASPREP

This maintains day/night differences within the schedules.

Longitude of carrier/15° = conversion factor (CF).

Sign of longitude will discern Sign of conversion factor (S(CF)).

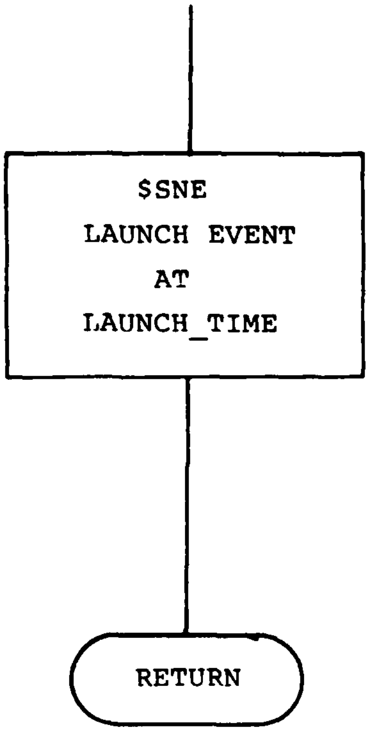
W = + E = -

LAUNCH_TIME = LAUNCH_TIME + (S(CF))

RTB = RTB + (S(CF))

Choose appropriate launch sequence by placing the pointer on the next sequence which time is greater than game time.

Call CASPREP Routine to evaluate scheduled time versus actual time required to prepare for launch added to game time and resolve conflicts.



Places launch event into the event stack with the pointer to the launch sequence desired.

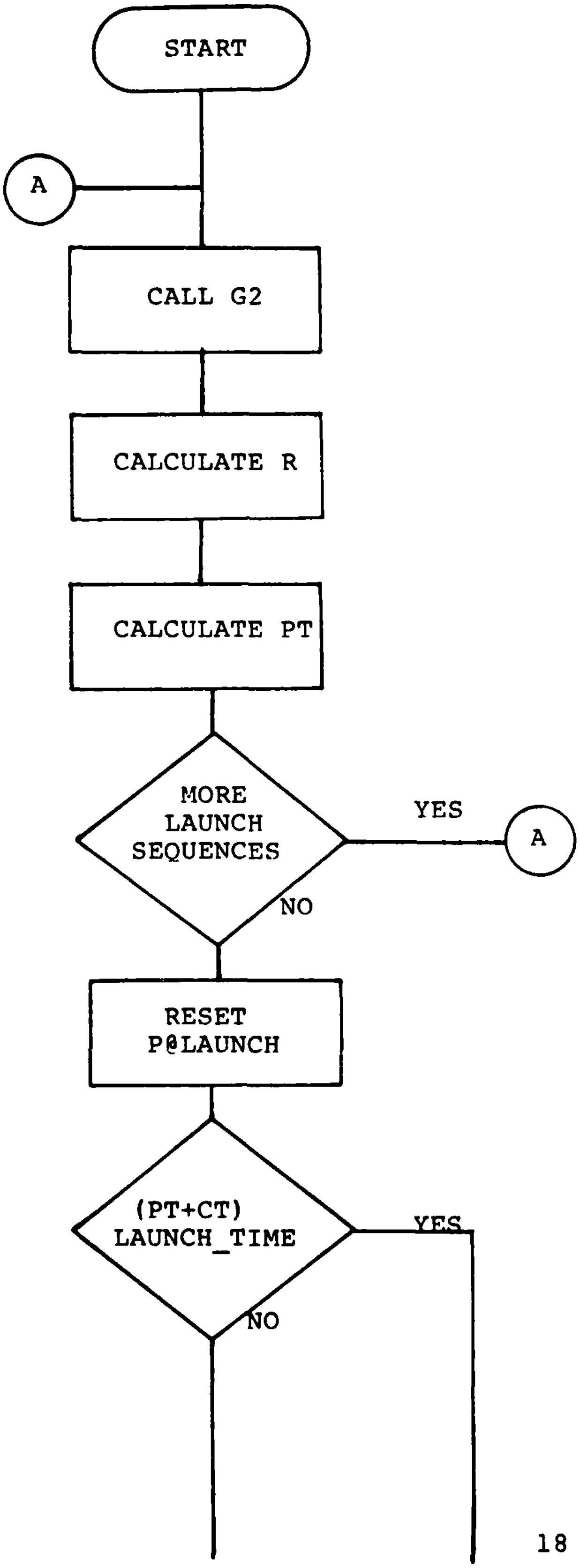
3.2 Preparation Time Routine

3.2.1 Description

To allow for an appropriate delav (caused bv planning, briefing, loading ordnance and fueling aircraft) between scheduling an event and the launching of aircraft, a routine called CASPREP was developed. In order to allow for an appropriate delay, a preparation time is calculated by selecting a random number between the minimum and maximum preparation times (ACPARM(I).PREPTIME_DIST(1), ACPARM(I).PREPTIME_DIST(2)) for each launch sequence based on the aircraft tvpe, and bv selecting the maximum preparation time.

CASPREP uses the preparation time to prepare the sequences that will be used in the launch, and compares it to the launch time scheduled for that event. If there is sufficient time available to prepare, then the event is placed in the event stack. However, if there is not sufficient time to prepare the crew and aircraft for that sequence of launches, this routine will advance the launch pointer (P@LAUNCH) to the next sequence of launches or delay the launch by scheduling the launch at the launch time plus the time needed for delav. To determine if the launch can be delayed, the routine compares the delav time with the slidetime (FLIGHT_SCHEDULE(*).SLIDETIME). If the delav time is larger than the slidetime for that schedule, then the preparation time is calculated for the next launch.

3.2.2 CASPREP FLOWCHART



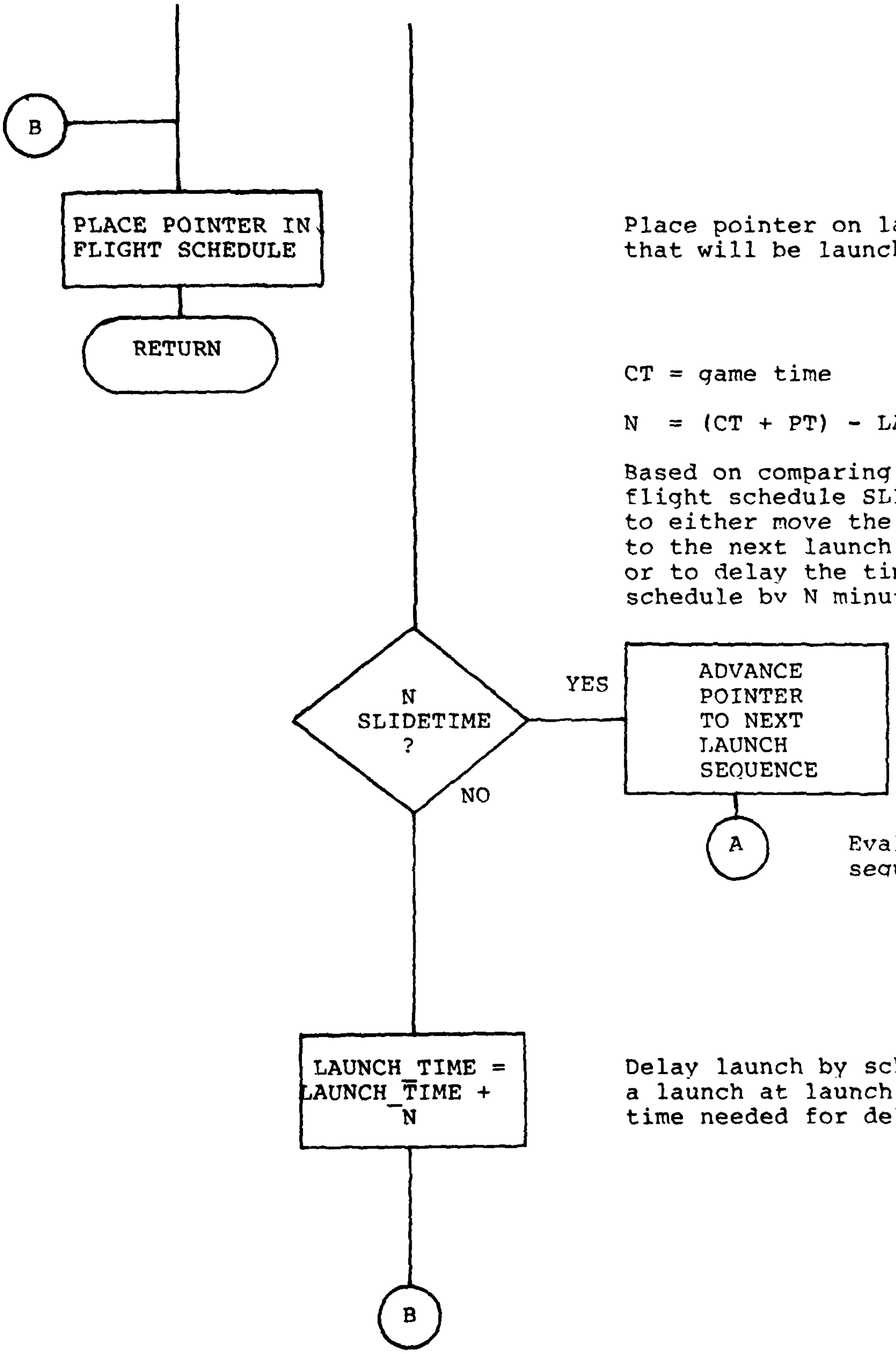
CASPREP called when a flight schedule is altered or a new flight schedule is developed

Cycle through the launch sequences that have a launch time equivalent to the execution time for the launch

Select and calculate a random number, R, to be between the minimum and maximum preparation time values (ACPARM(I).PREPTIME_DIST(1), ACPARM(I).PREPTIME_DIST(2)) PT is calculated by selecting the maximum of the calculated random values, R.

Reset P@LAUNCH to first pointer in group of launch sequences to be launched.

Compare time to prepare for launch with launch time.


B
PLACE POINTER IN FLIGHT SCHEDULE
RETURN
N SLIDETIME ?
YES
NO
ADVANCE POINTER TO NEXT LAUNCH SEQUENCE
A
LAUNCH_TIME = LAUNCH_TIME + N
B

Place pointer on launch event that will be launched next.

CT = game time

N = (CT + PT) - LAUNCH_TIME

Based on comparing N to a flight schedule SLIDETIME, to either move the pointer to the next launch sequence or to delay the times of the schedule bv N minutes.

Evaluate next seauence.

Delay launch by scheduling a launch at launch time plus time needed for delay.

4.0 AIRCRAFT CONTROLLING EVENTS

4.1 Description

CASAM portrays the establishment and execution of the entire range of carrier operations (e.g., alpha strike, cyclic operations, etc.). As described in Section 3.0, it provides the capability to schedule an aircraft. In the Launch Event aircraft are launched based on the launch sequences listed in the carrier's flight schedule and the availability of each type of aircraft shown on the carrier's Aircraft Status Board. Once the aircraft are launched, the ordnance which is usually either used or jettisoned to reduce the aircraft to landing weight can be automatically depleted from the carrier's ordnance stores. The ordnance items that would not be jettisoned are: air to air missiles, nuclear weapons, torpedoes, and sonobuoys. In addition to ordnance, a representative fuel load required for the mission to be flown can be depleted from the carrier's fuel supply. The airborne aircraft are then dispatched to accomplish their assigned mission.

When the Launch Event is complete, the Launch Complete Event is scheduled to station aircraft whose missions are equivalent to CAP, VS, or AEW, and to recover aircraft returning from engagements or stations. If the Launch Event sets a flag, LC_FLAG, then the Launch Complete event will only recover aircraft returning from their missions. A Recover Complete Event is then scheduled to send recovered aircraft to be refueled, rearmed and sent to maintenance if scheduled maintenance is required or if the aircraft suffered battle damage. The engagement modules will indicate those aircraft that are eliminated (i.e., damaged beyond ability to recover) during the engagement, and they will be added to the lost column in the Aircraft Status Board. The adjustments to the Aircraft Status Board will be made in the maintenance routine to indicate the unavailability of the above aircraft. A Maintenance Complete Event is scheduled in the maintenance routine to update the Status Board for aircraft returning from maintenance.

This Section explains the following components in detail:

- Status
- Aircraft Launch
- Launch Complete
- Recovery Complete
- Maintenance Complete
- Maintenance
- Station.

4.2 Status

Portions of CASAM will update the data required bv the other routines of CASAM, the engagement modules, and NNWS. Each carrier has an Aircraft Status Board (Figure 5) which is used to keep an up-to-date record of the aircraft for a specific carrier, and identifies the availability and location of that carrier's aircraft. The data structure for the Aircraft Status Board appears as follows:

```
DCL P@AC_STATUS_RECDS     POINTER
    AC_STATUS_RECDS_SIZE INITIAL(4/*64*/)
                         FIXED BINARY(15),
    1 AC_STATUS_RECDS BASED (P@AC_STATUS_RECDS),
      2 LINK             OFFSET(TESPACE),
      2 ACNAME           CHARACTER(8),
      2 ACTYPE           FIXED BINARY(15),
      2 ASSIGNED         FIXED BINARY(15),
      2 AVAILABLE(4)     FIXED BINARY(15),
      2 ACTIVE (8)       FIXED BINARY(15),
      2 TURNAROUND (2)   FIXED BINARY(15),
      2 LOST             FIXED BINARY(15),
      2 PAD              CHARACTER(22);
```

It is planned that most engagement modules will require information from only the Aircraft Status Board to obtain the number of aircraft assigned to enter into an engagement; and information from the flight schedule to obtain ordnance loads on those aircraft. The notable exception to this plan is the Naval Air Defense Simulation (Reference 1). Data will be obtained for NADS from the NNWS checkpoint files by the NADS interface program.

CARRIER 66 AIRCRAFT
STATUS BOARD

	A6	A7	F14	EA6	E2	SH3	S3	KA6
ASSIGNED	9	18	24	4	4	8	10	4
MAINTENANCE (TURNAROUND(2))	0	2	7	1	1	0	2	1
REFUEL_REARM (TURNAROUND(1))	2	4	3	1	1			1
CAP (ACTIVE(4))			3					
VS (ACTIVE(5))						2	2	
AEW (ACTIVE(6))					2			
AC_LANDATTACK (ACTIVE(7))								
AC_SEATTACK (ACTIVE(8))	2	4						
EW_ESCORT (ACTIVE(2))								
EW_STATION (ACTIVE(3))					1			
TANKER (ACTIVE (1))								1
DLI_N1 (AVAILABLE(9))			1					
DLI_N2 (AVAILABLE(10))			1					
DLI_N3 (AVAILABLE(11))								
LOST								
DLI_N4 (AVAILABLE(4))	5	8	9	1	0	6	6	3

FIGURE 5

Each time an aircraft status is altered, an update will occur to the appropriate board to maintain an up-to-date record of this data. Therefore, when an aircraft returns to an available status, an availability update will occur.

4.3 Launch Event

4.3.1 Description

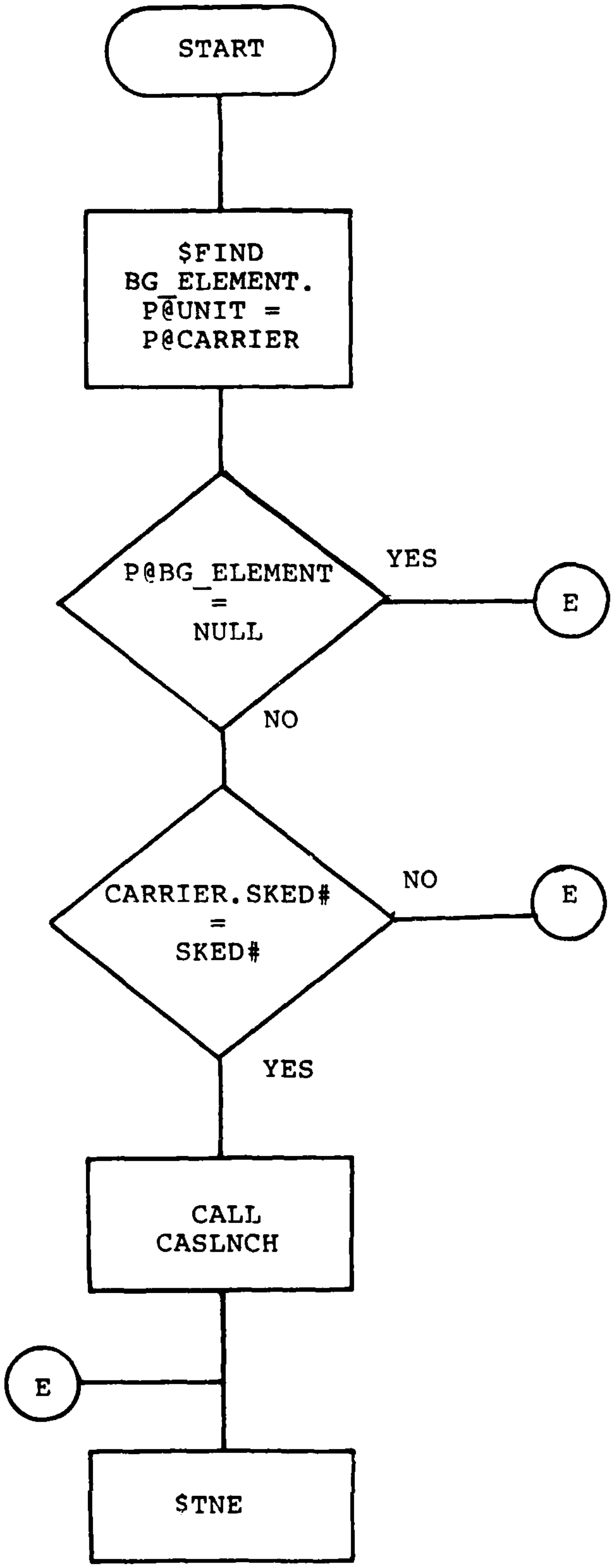
The Launch Event initiates the launching sequence of the carrier aircraft. There are two ways to schedule a Launch Event. If this is a new schedule, then the event is scheduled by the Flight Schedule Routine. Otherwise, if this is an ongoing schedule, then the next launch event is scheduled by the Launch Routine (CASLNCH).

To perform the Launch Event, a pointer to a battlegroup, P@BGROUP, and a pointer to a carrier, P@CARRIER, are necessary to determine if the carrier has been destroyed prior to executing the event. If the carrier has been destroyed, then the next event is taken from the event queue. The number of towed arrays and a disposition selection number are passed to the Launch Event to pass to the Launch Complete Event to be utilized in the Station Routine, CASSTAT. The disposition selection number is also used to obtain a schedule number from the disposition selection table.

```
DCL  1 ENOTE60           BASED(P@ENOTE),
       2 SYSTEM          CHARACTER(16),
       2 P@CARRIER       OFFSET(TESPACE),
       2 P@LAUNCH        OFFSET(TESPACE),
       2 P@BGROUP        OFFSET(TESPACE),
       2 #TOWED_ARRAYS   FIXED BINARY(15),
       2 DISP_ELEMENT    FIXED BINARY(15),
       2 PAD             CHARACTER(6).
```

If the schedule number passed is for the execution of the launch event, then the Launch Event calls, CASLNCH. This routine will cycle through a group of launch sequences and launch aircraft based on the launch time. Once the cycle is completed, the routine will schedule a Launch Complete Event and another Launch Event.

4.3.2 LAUNCH EVENT FLOWCHART



Using the $FIND macro find the BG_ELEMENT contained in the Battlegroup composition list which has a pointer to the unit, P@UNIT, which is equivalent to the pointer P@CARRIER passed. This determines if the carrier was destroyed prior to executing the event.

Validate event to ascertain if the event schedule number contained in the Disposition Selection Table (DISPSELTABLE (DISP_ELEMENT).INDEX.SKED#).

Call the launch routine to perform aircraft launches based on the time the event is executed.

Take the next event from the queue.

4.3.3 Launch Routine

The Launch Routine, CASLNCH, is called by the Launch Event to launch aircraft based on the Launch Time. The routine will cycle the group of launch sequences and launch aircraft with a launch time equal to the current gametime. Figure 6 is an example flight schedule of the launch sequences contained in a carrier's flight schedule. The launch sequences are filled in order of priority (the first launch sequence takes priority over the following sequences).

The routine will determine the number of aircraft available for the launch, and update the availability status in the Status Board for each aircraft type utilized. To determine the number of aircraft that will be aborted from the launch, compare the abort rate of the aircraft type to be launched (ACPARM(I).ABORT_RATE) with a random number selected between 0 and 1. The aircraft that are aborted will be sent to maintenance. Otherwise, if the launch sequence has an aircraft mission equivalent to a CAP, VS, or AEW then the Launch Event will call the subroutines UNINIT, and CREATE to create an aircraft unit and its associated battlegroup element, BG_ELEMENT. These are station aircraft which are utilized to help in the defense of the battle group. The aircraft declaration which contains information that is pertinent for the aircraft units appears as follows:

```
DCL P@AIRCRAFT              OFFSET(TESPACE),
    1 AIRCRAFT              BASED(P@AIRCRAFT),
      2 UNIT                CHARACTER(400),
      2 P@CARRIER           OFFSET(TESPACE),
      2 RTB                 FIXED BINARY(31),
      2 ACTYPE              FIXED BINARY(15),
      2 MISSION             FIXED BINARY(15),
      2 ORDNANCE            FIXED BINARY(15),
      2 PAD                 CHARACTER(98).
```

Launch Time	Type Aircraft	# Air- craft	Mission	Ord.	RTB
0360	F14	3	CAP	4 PHOENIX	0450
0360	F14	1	DLI_N1	4 PHOENIX	0450
0360	A7	4	DLI_N2	4 PHOENIX	0450
0360	KA7	1	TANKER	N/A	0450
0360	A6	2	STRIKE	28 MK 82	0450
0360	KA6D	1	DLI_N1	N/A	0450
0360	EA6	1	EW	4 EW PODS	0540
0360	E2C	1	AEW	N/A	0630
0360	S3A	1	ASW	N/A	0630
0360	SH3D	1	ASW	N/A	0630
0450	F14	3	CAP	4 PHOENIX	0540
0450	F14	1	DLI_N1	4 PHOENIX	0540
0450	F14	1	DLI_N2	4 PHOENIX	0540
0450	A7	4	STRIKE	12 MK 82	0630
0450	A6	2	STRIKE	28 MK 82	0540
0450	KA6D	1	TANKER	N/A	0540
0450	S3A	1	ASW	N/A	0720
0450	SH3D	1	ASW	N/A	0630
0540			RECOVER		0630

etc.

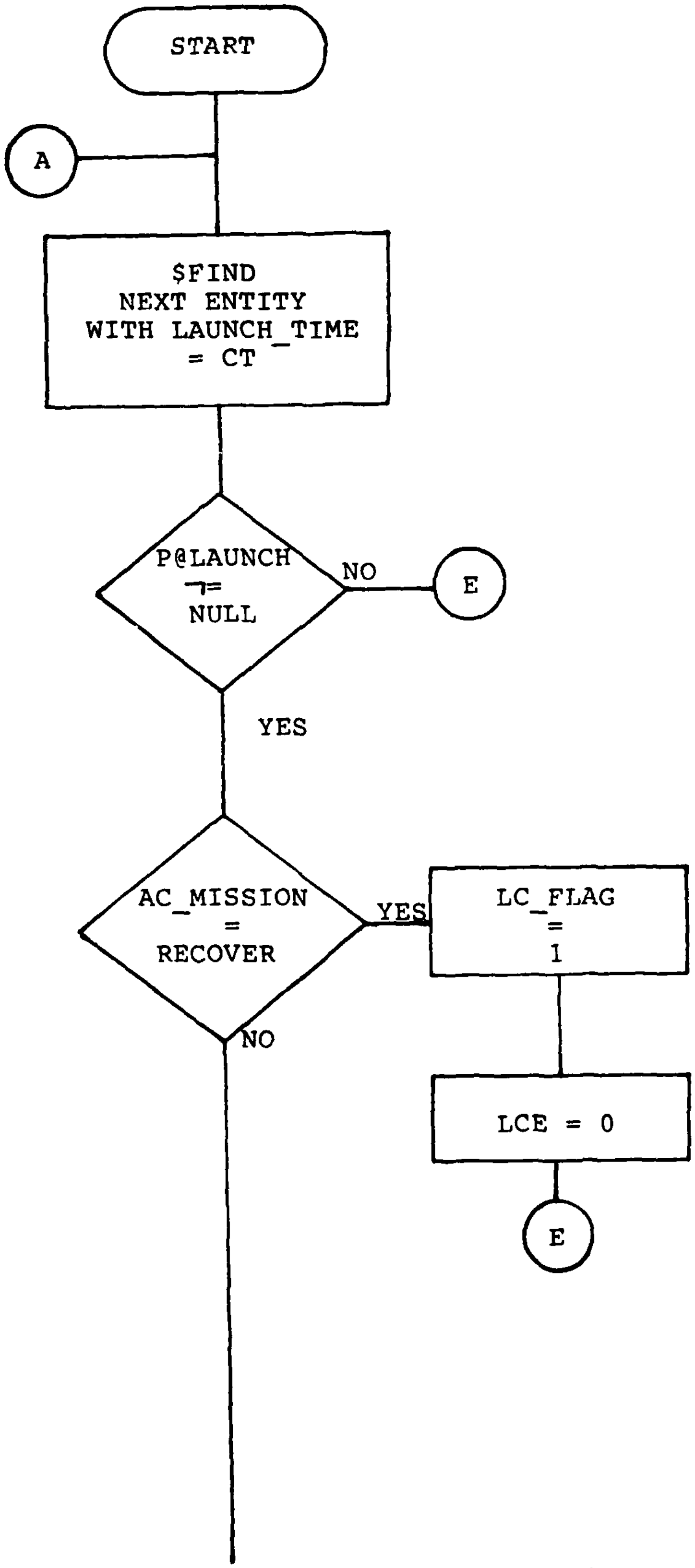
FIGURE 6

If the launch sequence contains an aircraft mission equivalent to a AC_SEATTACK or AC_LANDATTACK, then the Launch Routine creates an engage entity for engagement aircraft, and links it to the Engagement LINK List contained on the carrier. The Engage declaration appears as follows:

```
DCL P@ENGAGE                  OFFSET(TESPACE),
    ENGAGE_SIZE               FIXED BINARY(15),
      1 ENGAGE                BASED(P@ENGAGE),
      2 LINK                  OFFSET(TESPACE),
      2 ACTYPE                FIXED BINARY(15),
      2 #ACTYPE               FIXED BINARY(15),
      2 MISSION               FIXED BINARY(15),
      2 ORDNANCE              FIXED BINARY(15),
      2 RTB                   FIXED BINARY(31).
```

To calculate the execution time for Launch Complete Event, LCE, take the sum of the number of aircraft of each type to be launched and multiply by its launch rate. When LCE is calculated, compare it with the time contained in the battle group (BGROUP.LNCHCMP_RCVRCMP_TIME) to determine when the last Launch Complete Event or Recovery Complete Event will be executed. If LCE exceeds the value contained in BGROUP.LNCHCMP.RCVRCMP_TIME, then replace the value with LCE. If the aircraft mission is to perform a recovery, LAUNCH.AC_MISSION=RECOVER, then set a flag to true, to the Launch Complete Event.

4.3.4 CASLNCH FLOWCHART



The Launch Routine is called by the Launch Event.

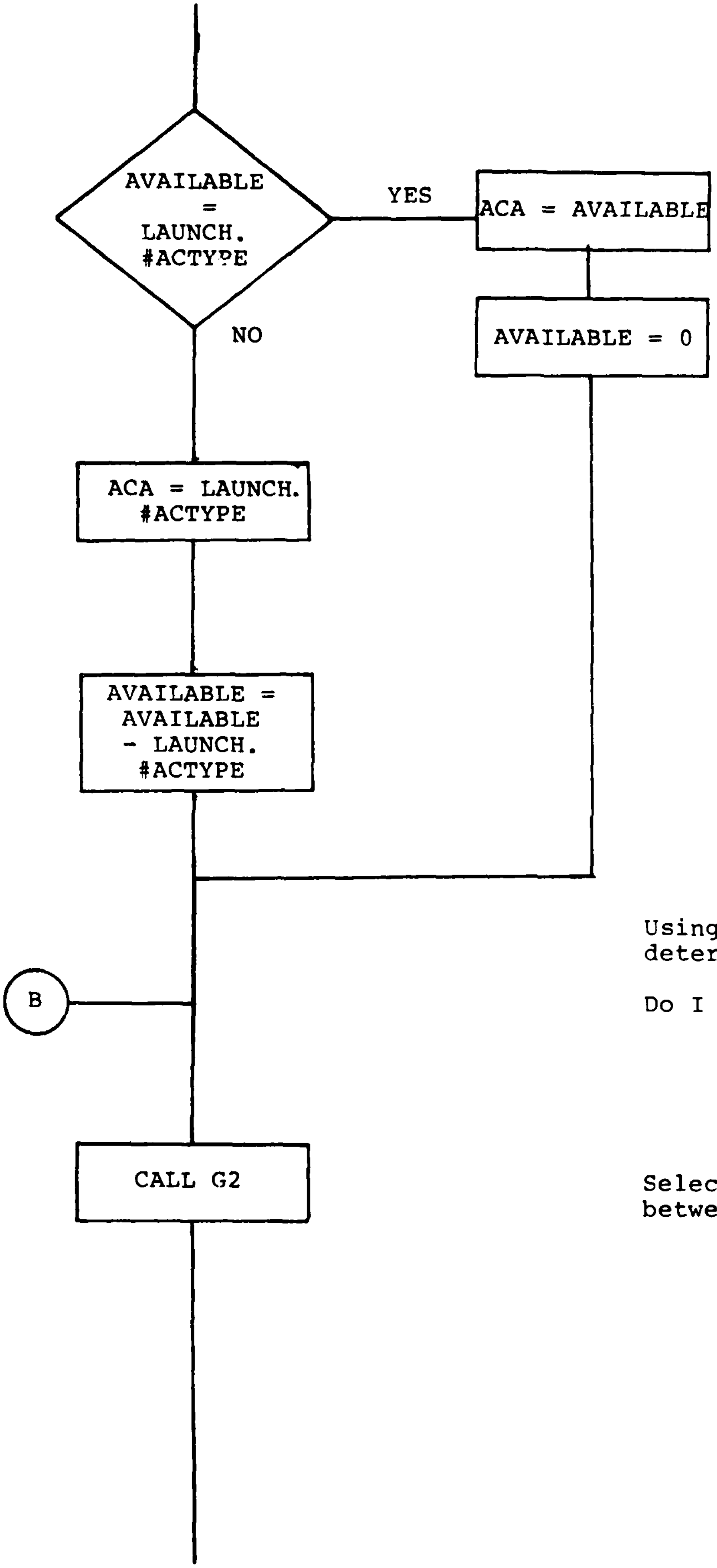
Do until launch time contained in launch sequence does not equal current game time.

Find the next entity in a carrier's flight schedule link list, the launch sequence's launch time equal to the game time.

If there is another launch sequence contained in the list, then launch aircraft contained in sequence.

Set a launch complete flag to indicate the recovery of aircraft.

Set the launch complete time for scheduling a launch complete event to zero.

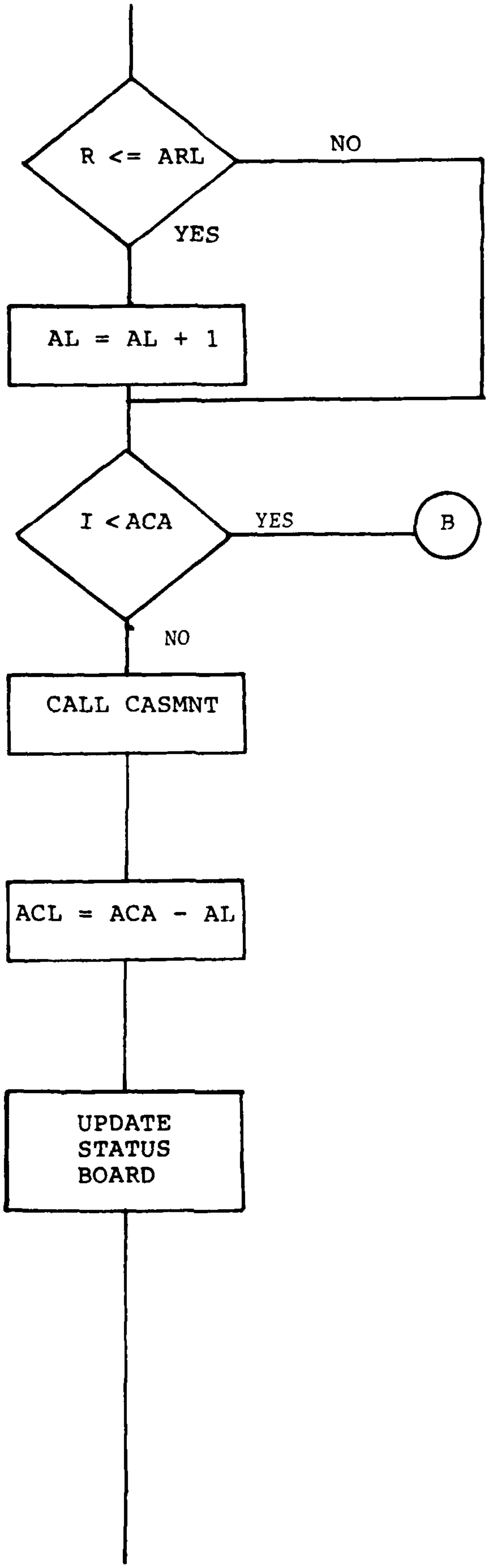

AVAILABLE = LAUNCH. #ACTYPE
YES
ACA = AVAILABLE
AVAILABLE = 0
NO
ACA = LAUNCH. #ACTYPE
AVAILABLE = AVAILABLE - LAUNCH. #ACTYPE
B
CALL G2

Update availability in the Status Board for a certain aircraft type

Using a random distribution determine

Do I = to ACA;

Select a random number, R between 0 and 1

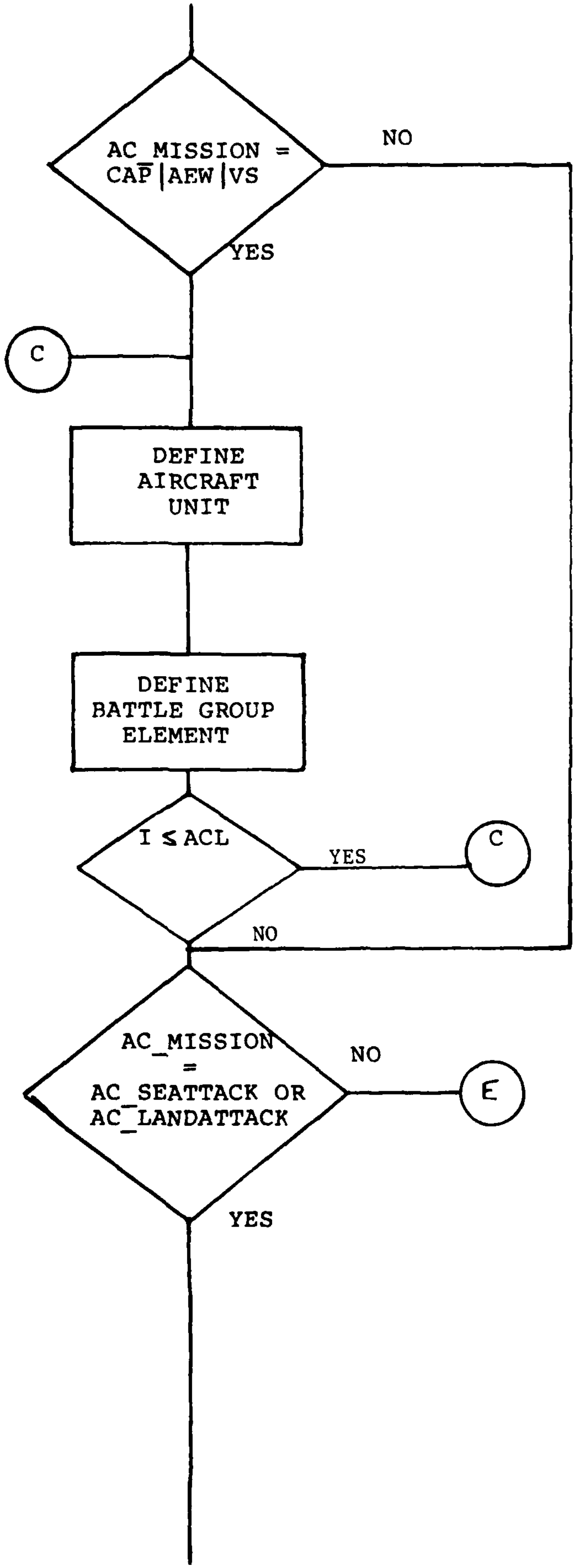


ARL = ABORT RATE for a specific aircraft type

AL = number of aircraft aborted

Send aborted aircraft to maintenance.

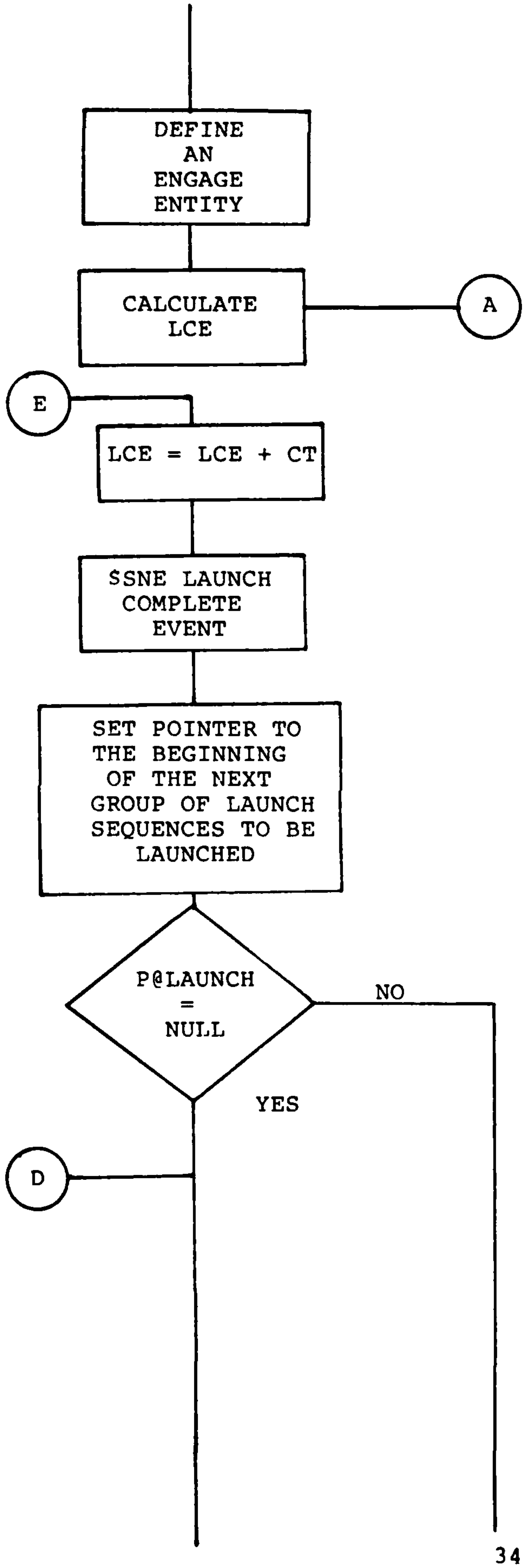
ACL = number of aircraft to be launched



DO I = 1 to ACL

Using the UNINIT and the Create subroutines allocate dynamically an entity in TESPACE for both the aircraft unit and its BG_ELEMENT. Then, use the $LINK macro to link to the appropriate BG_ELEMENT. Each aircraft unit will contain the following attributes:

: Pointer to the carrier
: RTB time
: Launch time
  (Additional Unit Data)


DEFINE AN ENGAGE ENTITY
CALCULATE LCE
A
E
LCE = LCE + CT
$SNE LAUNCH COMPLETE EVENT
SET POINTER TO THE BEGINNING OF THE NEXT GROUP OF LAUNCH SEQUENCES TO BE LAUNCHED
P@LAUNCH = NULL
NO
YES
D

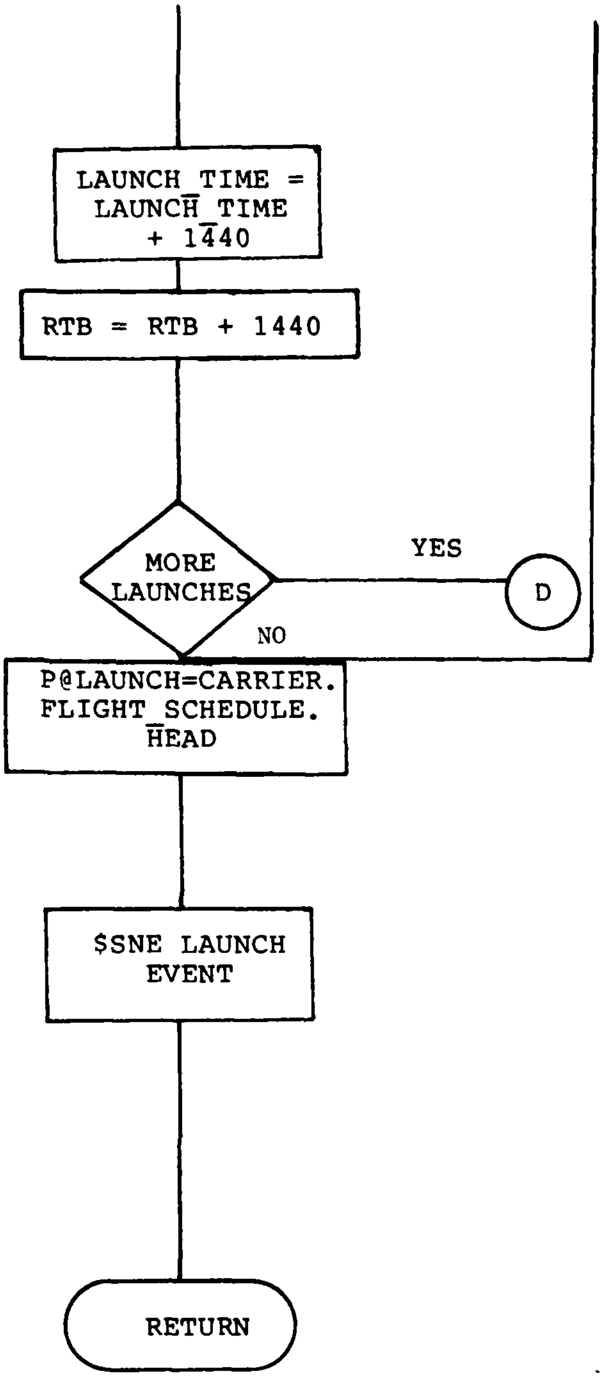
Using the Create subroutine and $LINK macro allocate dynamically an engage entity in TESPACE and link it to the engagement link list (CARRIER. ENGAGEMENT).

LCE = LCE + ROUND(ACL*LR,0)
 LR = LAUNCH_RATE

Calculate the execution time for the Launch Complete Event.

Schedule a Launch Complete Event.

Cycle through launch link list



Add one day to the launch times and RTB times

Schedule the next launch event

4.4 Launch Complete Event

4.4.1 Description

The Launch Complete Event and the Recover Event are incorporated, so that when the Launch Complete Event is scheduled, it will perform only the recovery of aircraft if a flag is set to be true, otherwise it will perform both the launch complete and the recovery of aircraft.

To perform the Launch Complete Event, a pointer to a battle group, P@BGROUP, and a pointer to a carrier, P@CARRIER, are necessary to determine if the carrier has been destroyed prior to executing the event. If the carrier has been destroyed, then the returning aircraft are destroyed, else the event calls the Launch Complete Routine, CASLCMP. The LC_FLAG is passed through the scheduling of the event, in order to determine if a recovery occurs (LC_FLAG = '1'B) or if a launch complete and recovery occurs (LC_FLAG = '0'B). The number of towed arrays and the disposition number are passed to the event, to be utilized in the Station Routine which is called by the Launch Complete Routine. The event notice declaration for the Launch Complete Event appears as follows:

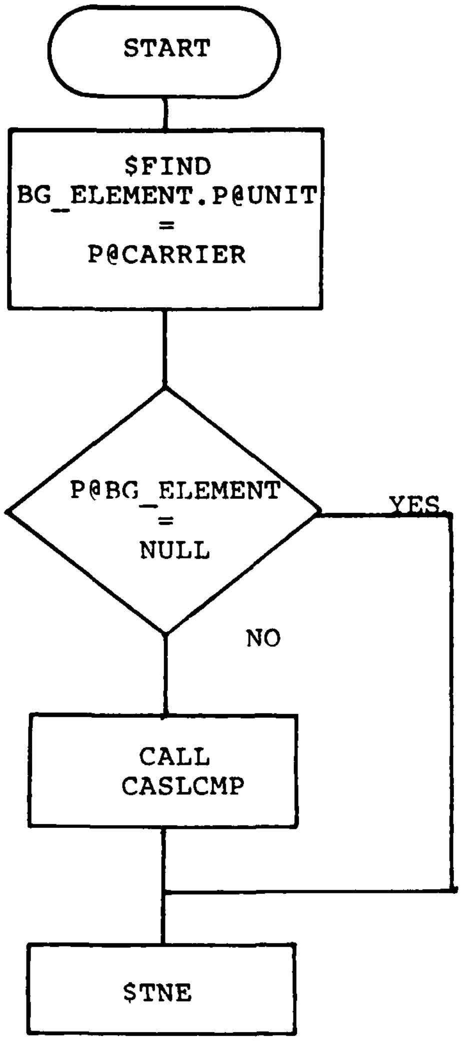
```
DCL  1  ENOTE61           BASED(P@ENOTE),
        2 SYSTEM          CHARACTER(16),
        2 P@CARRIER       OFFSET(TESPACE),
        2 P@BGROUP        OFFSET(TESPACE),
        2 LC_FLAG         BIT(1),
        2 #TOWED_ARRAYS   FIXED BINARY(15),
        2 DISP_ELEMENT    FIXED BINARY(15),
        2 PAD             CHARACTER(10);
```

The event will record the identity of the aircraft launched for a flight summary report, if required, and at this time reduces the carrier's aircraft ordnance bv the amount of jettisonable ordnance on board the launching aircraft.

Non-jettisonable ordnance are as follows:

1. Torpedoes
2. Nuclear Weapons
3. Air to Air Missiles
4. Sonobuoys.

4.4.2 LAUNCH COMPLETE EVENT FLOWCHART



Using $FIND macro find a pointer to a BG_ELEMENT (battle group element) that has a pointer to the unit, P@UNIT, which is equivalent to the pointer P@CARRIER passed through the event. This determines if the carrier was destroyed prior to executing the event.

Call the Launch Complete Routine to calculate the number of aircraft returning from stations and engagements, and to station aircraft.

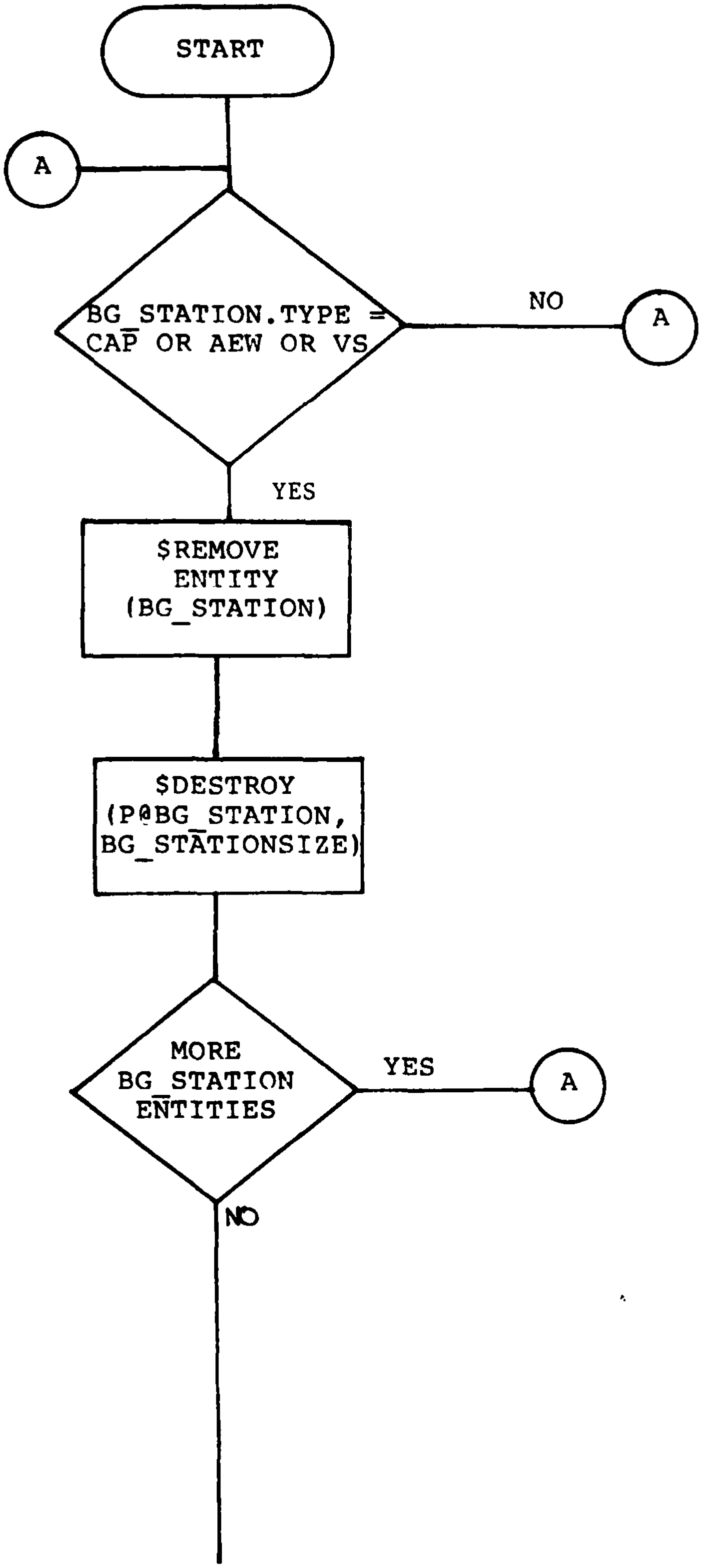
Take the next event from the event queue

4.4.3 Launch Complete Routine

The Launch Complete Routine, CASLCMP, is called by the Launch Complete Event to either perform a launch complete, or recoverv of aircraft. Based on the flag passed to the routine, if the flag is set to true, then a recovery routine is performed to calculate the number of aircraft returning from stations and engagements. Otherwise, the Launch Complete routine is performed to station aircraft that have an aircraft mission equivalent to CAP, VS, or AEW, and it also calculates the number of aircraft returning from stations and engagement modules.

Based on the number of aircraft returning and the recoverv rate for each aircraft, an execution time is calculated for the Recover Complete Event. This execution time is compared with the time contained in the battle group (BGROUP.LNCHCMP_RCVRCMP_TIME) to determine when the last Launch Complete or Recoverv Complete Event will be executed.

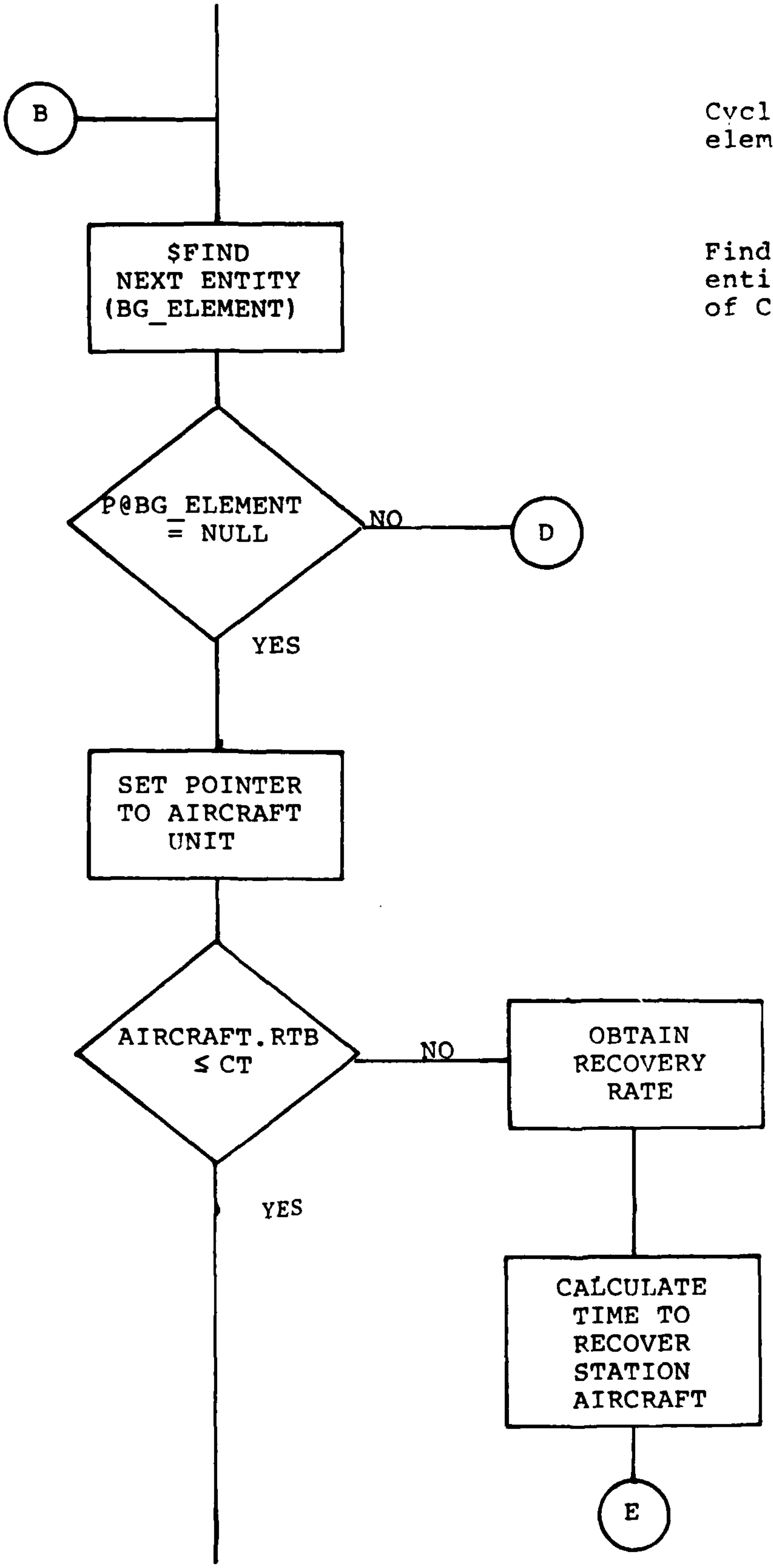
4.4.4 CASLCMP FLOWCHART


START
A
BG_STATION.TYPE = CAP OR AEW OR VS
NO
A
YES
$REMOVE ENTITY (BG_STATION)
$DESTROY (P@BG_STATION, BG_STATIONSIZE)
MORE BG_STATION ENTITIES
YES
A
NO

Cycle through the battle group station list.

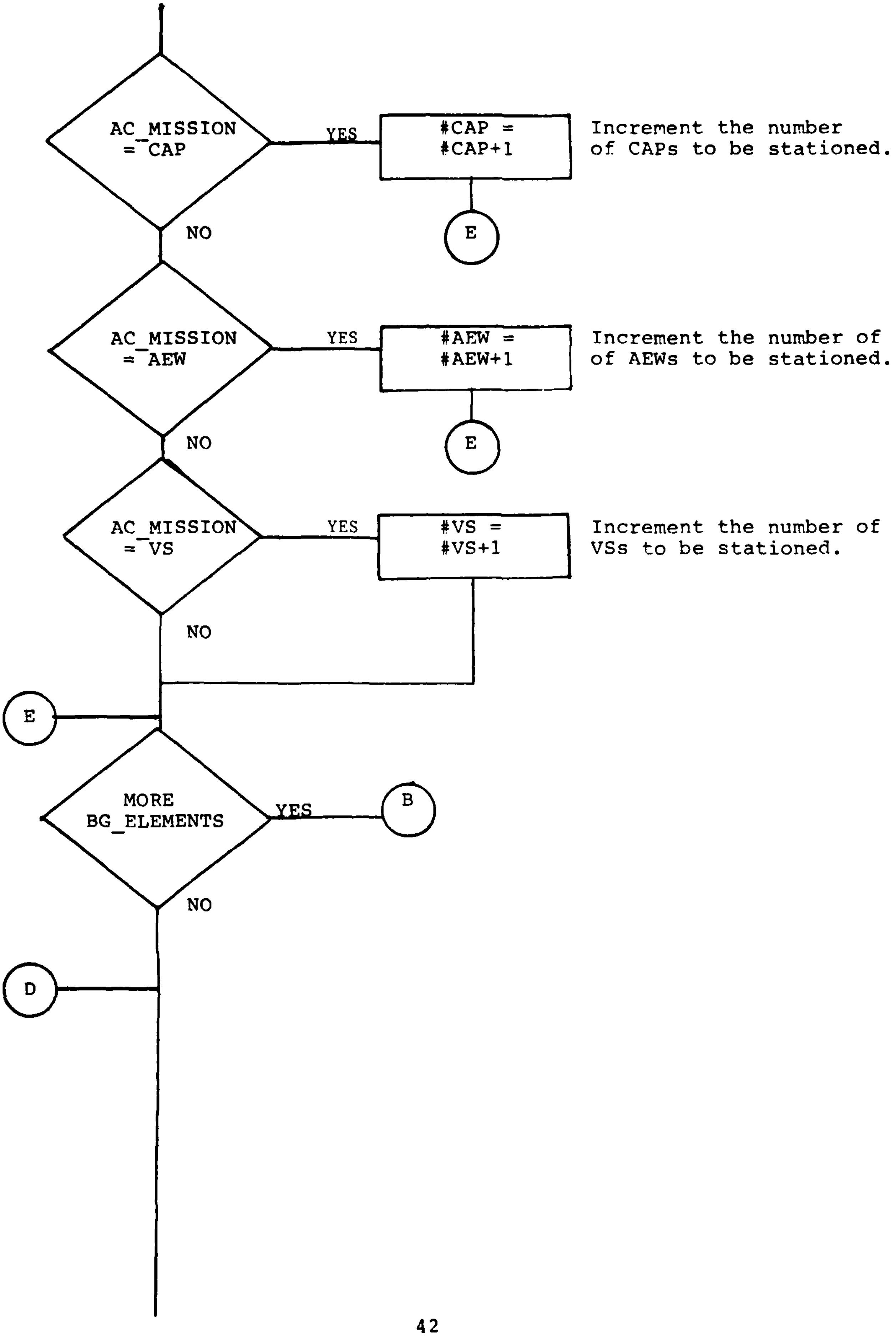
If the battle group station entity contains a station type equal to CAP, VS, or AEW then remove the station entity, else go to the next entity in the list.

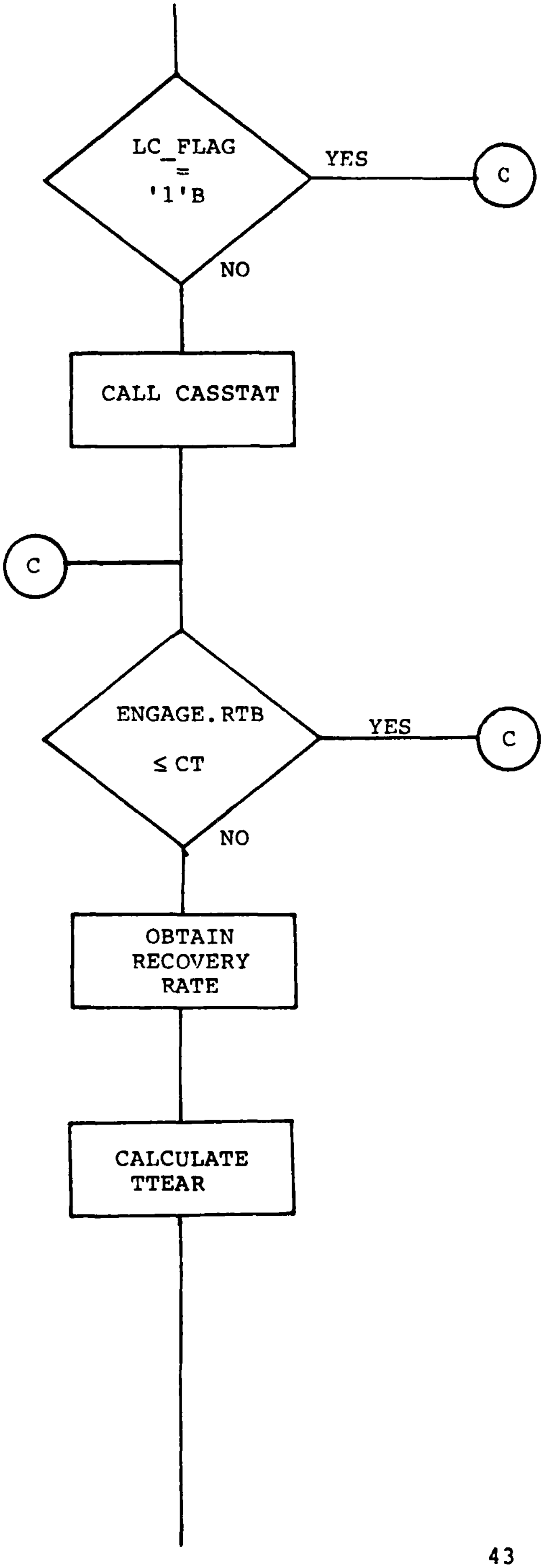
Use the $REMOVE and $DESTROY macros to remove a station entity from the battle group station list.



Cvcle through the battle group element list.

Find a battle group element entitv contains a station type of CAP, VS or AEW.


AC_MISSION = CAP
YES
#CAP = #CAP+1
Increment the number of CAPs to be stationed.
E
NO
AC_MISSION = AEW
YES
#AEW = #AEW+1
Increment the number of of AEWs to be stationed.
E
NO
AC_MISSION = VS
YES
#VS = #VS+1
Increment the number of VSs to be stationed.
NO
E
MORE BG_ELEMENTS
YES
B
NO
D



Determine if Routine is only to recover aircraft

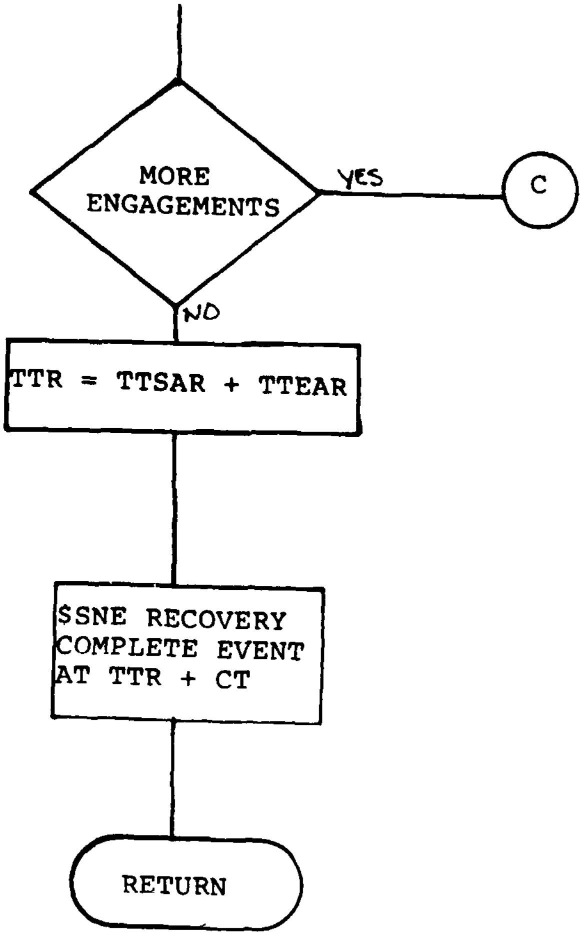
Call subroutine CASSTAT to create stations for aircraft with missions equivalent to CAP, VS, or AEW.

Cycle through engagement link list (CARRIER.ENGAGEMENT)

If engagement aircraft is not ready to return then go to next engagement.

Obtain recovery rate for specified ACTYPE

TTEAR = Total Time for Engagement Aircraft



TTR = Total time to recover all aircraft

Schedule a recovery complete event at TTR + CT

4.5 Recovery Complete Event

4.5.1 Description

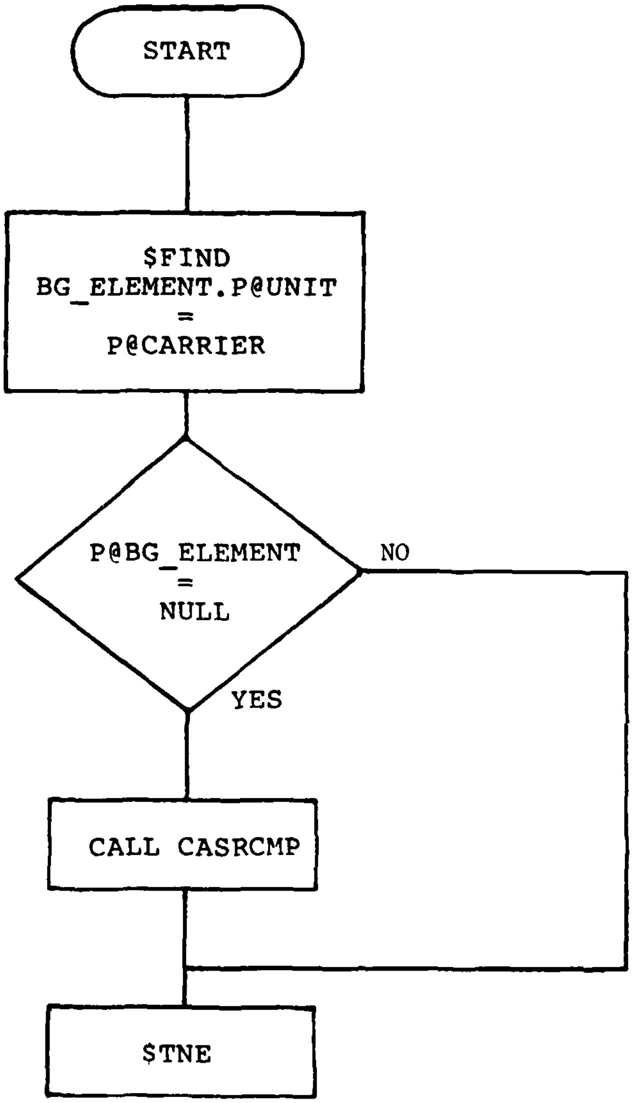
The Recovery Complete Event calls the Recovery Complete Routine, CASRCMP, to return aircraft from stations and engagement modules.

To perform the Recovery Complete Event, a pointer to a battle group, P@BGROUP, and a pointer to a carrier, P@CARRIER, are necessary to determine if the carrier has been destroyed prior to executing the event. If the carrier has been destroyed then get the next event from the event queue.

The event notice declaration for the Recovery Complete Event appears as follows:

```
DCL  1 ENOTE62              BASED(P@ENOTE),
       2 SYSTEM             CHARACTER(16),
       2 P@CARRIER          OFFSET(TESPACE),
       2 P@BGROUP           OFFSET(TESPACE),
       2 PAD                CHARACTER(12).
```

4.5.2 RECOVERY COMPLETE EVENT FLOWCHART



Using the $FIND macro find a point to be a BG_ELEMENT (battle group element) that has a pointer to the unit, P@UNIT, which is equivalent to the pointer P@CARRIER passed through the event. This determines if the carrier was destroyed prior to execution of the event.

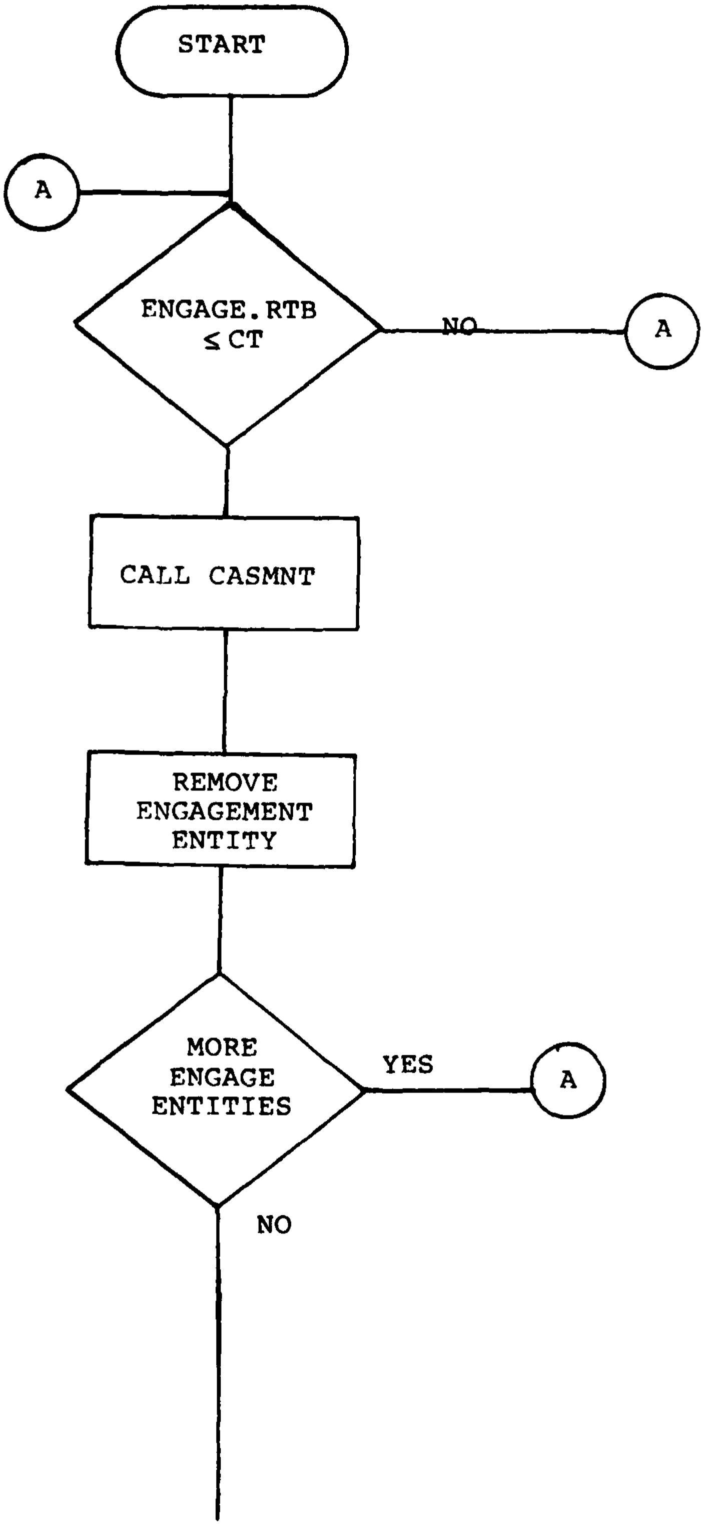
Call the Recovery Complete Routine.

Take the next event from the event queue.

4.5.3 Recovery Complete Routine

The Recovery Complete Routine, CASRCMP, queries both the composition list and the engagement list for aircraft returning from stations and engagement modules. Returning aircraft are refueled, rearmed, and sent to maintenance if scheduled maintenance is required, or if the aircraft suffered battle damage via the CASMNT Routine.

4.5.4 CASRCMP FLOWCHART

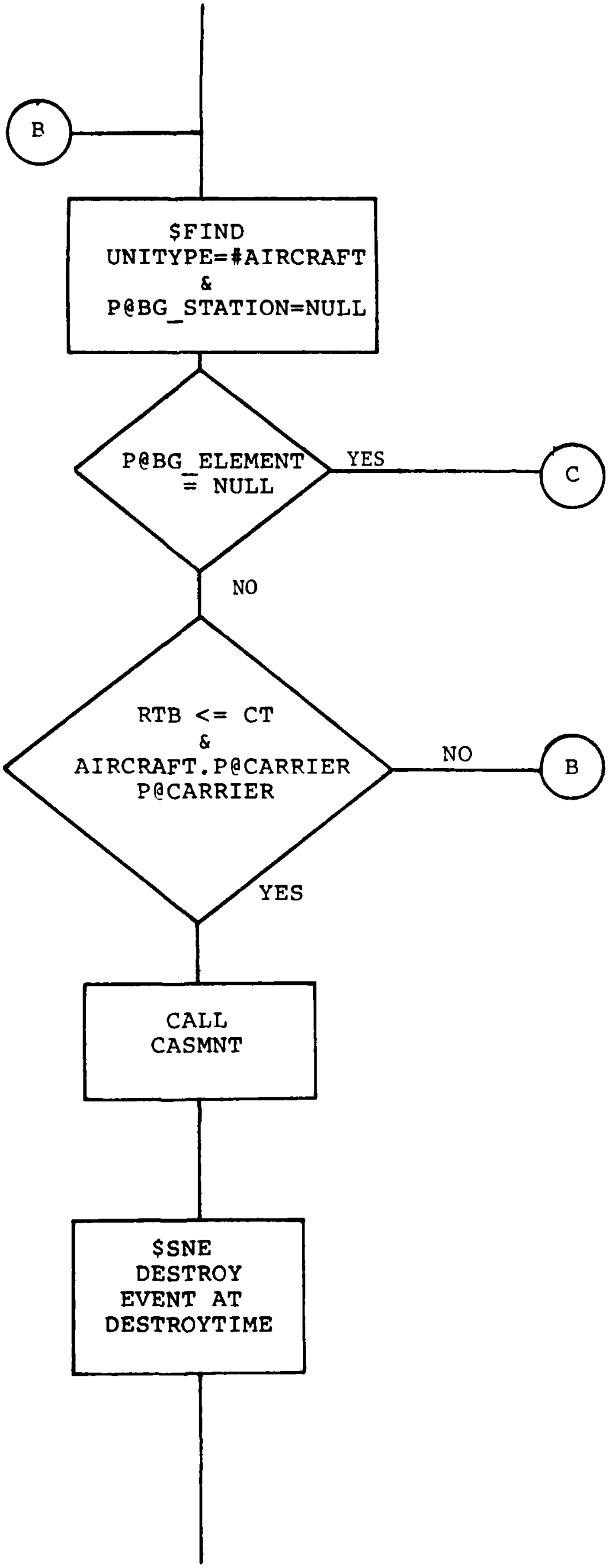


Cycle through the engagement link list

Determine if engagement aircraft are ready to return

Send engagement aircraft returning to maintenance.

Using $REMOVE and $DESTROY macros removing the engagement entity from the engagement link list



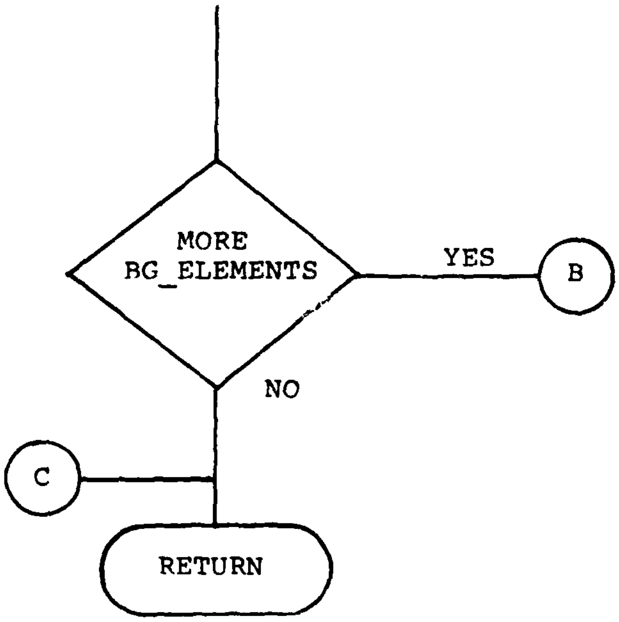
Cycle through Battlegroup Composition list

Using the $FIND macro, find the next BG_ELEMENT entity which points to an aircraft unit, and does not point to a BG_STATION entity.

Determin if an aircraft unit contains an RTB that is less than or equal to CT, and if the aircraft unit contains a carrier pointer equivalent to the pointer passed to the routine.

Send station aircraft to maintenance

Schedule the Destroy Event (EN13) to remove an aircraft unit from the unit list


MORE BG_ELEMENTS
YES
B
NO
C
RETURN

4.6 Maintenance Complete Event

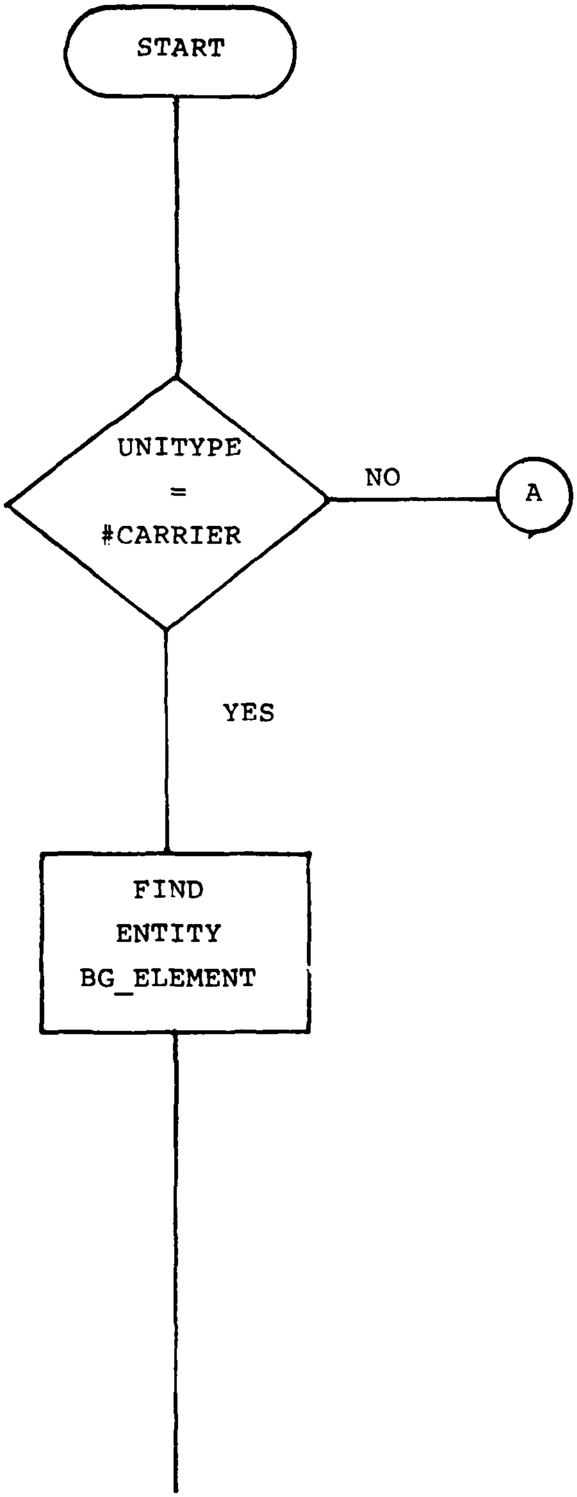
4.6.1 Description

The Maintenance Complete Event is scheduled to update the Aircraft Status Board for returning aircraft units or air raid elements. If an aircraft unit is returning from maintenance then the carrier Status Board is updated. If an air raid element is returning from an engagement then the air field Status Board is updated.

To perform this event a pointer to a carrier or an air field unit, P@UNIT, is required to specify which Status Board is being updated. If the returning unit is an aircraft, then a pointer to the battle group, P@BGROUP, is also needed to indicate if the carrier has been destroyed prior to executing the event. The event notice declaration for the event appears as follows:

```
DCL 1 ENOTE63              BASED(P@ENOTE),
      2 SYSTEM             CHARACTER(16),
      2 P@UNIT             OFFSET(TESPACE),
      2 P@BGROUP           OFFSET(TESPACE),
      2 ACTYPE             FIXED BINARY(15),
      2 STAT               FIXED BINARY(15),
      2 PAD                CHARACTER(12).
```

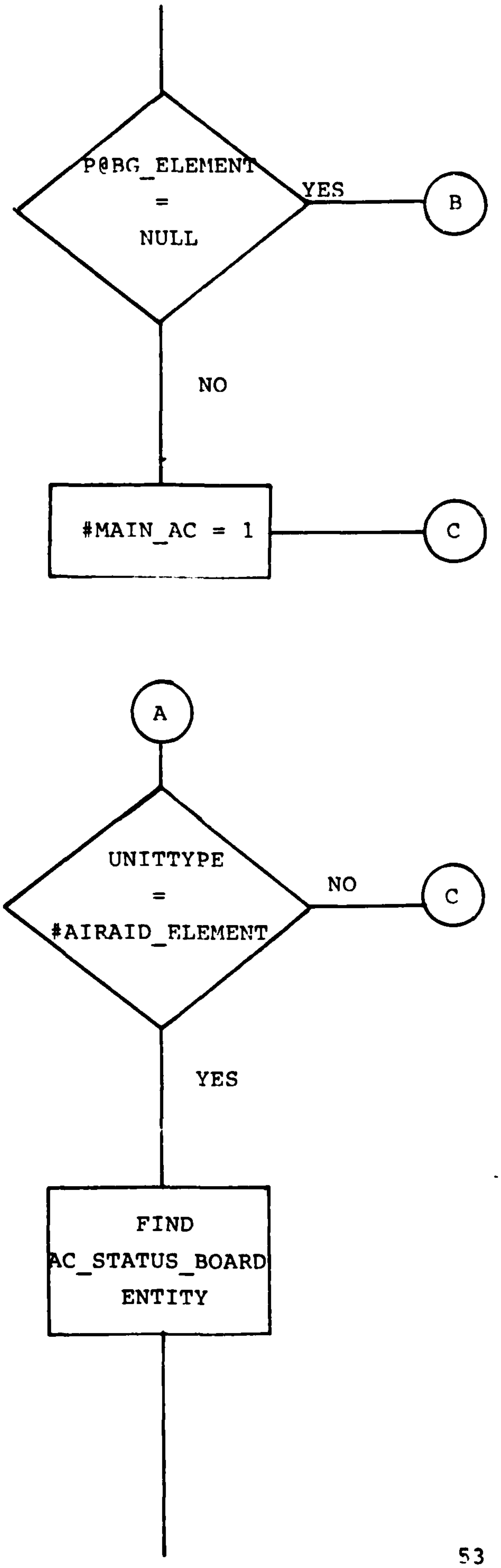
4.6.2 MAINTENANCE COMPLETE FLOWCHART



Scheduled by the Maintenance Routine and the Launch Routine.

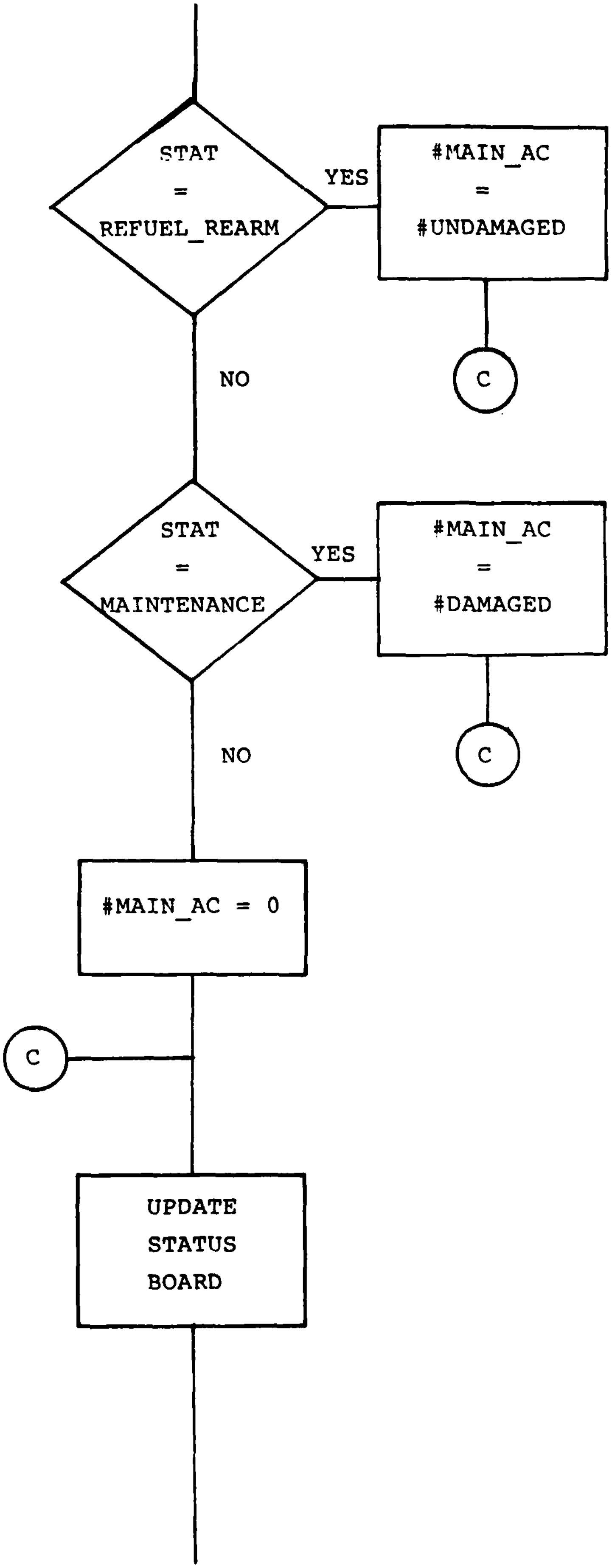
If the unit passed to the event is a carrier.

Search the battle group composition for a pointer to the carrier equal to the pointer passed to the event.



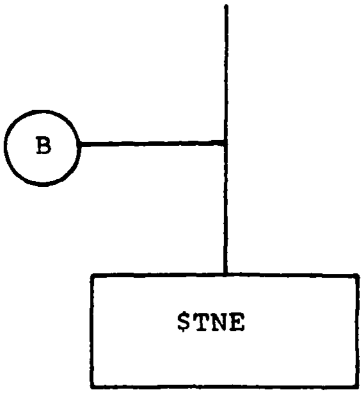
Set the number of aircraft returning from maintenance.

Set pointer to Status



Set the number of air raid elements returning from maintenance eoual to the number of undamaged elements.

Set the number of air raid elements returning from maintenance equal to the number of damaged elements.



Take next event from event queue.

4.7 Maintenance Routine

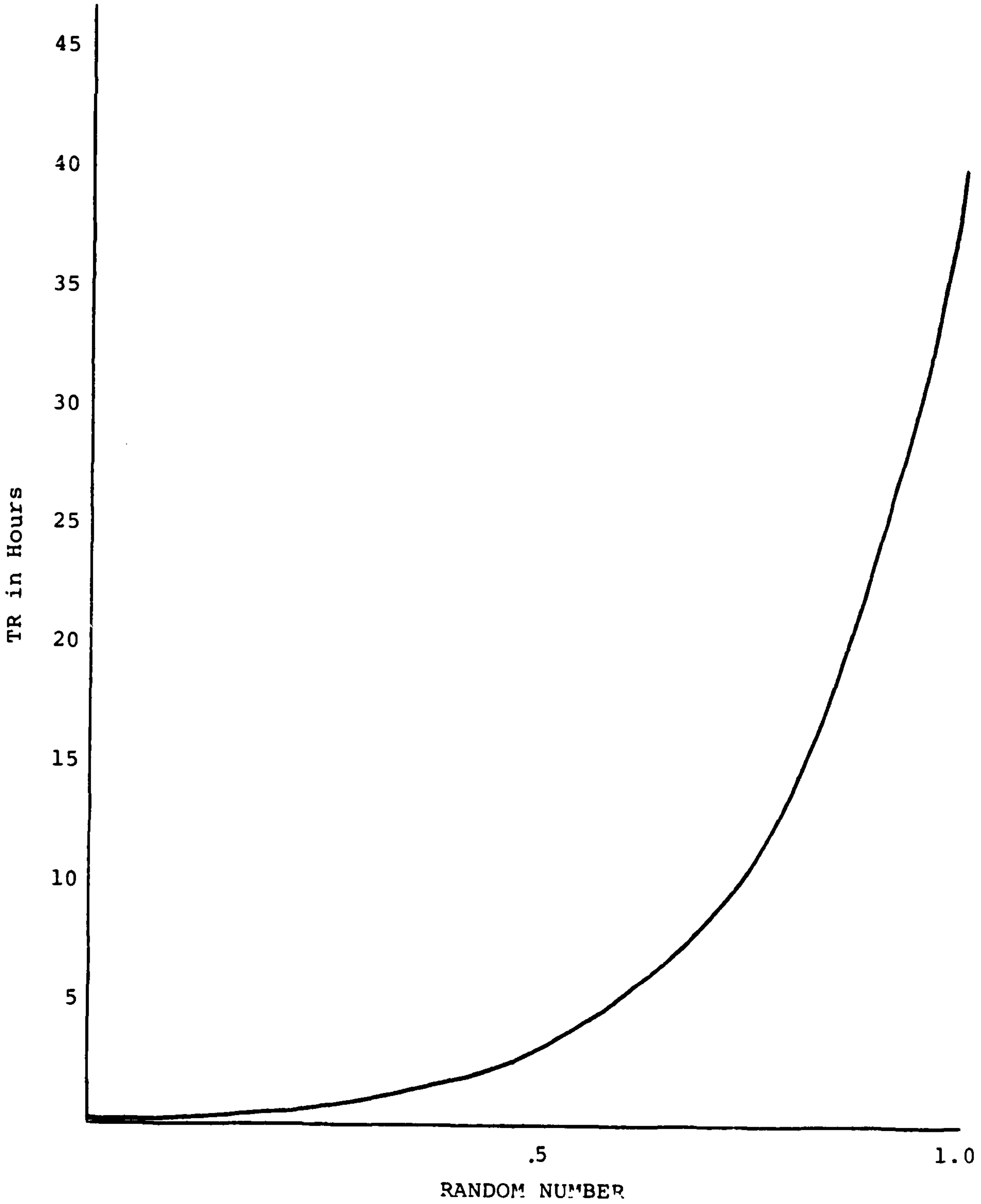
4.7.1 Description

The Maintenance Routine calculates the time at which an aircraft will return to the available status. All aircraft whose launch is aborted, or are recovered from a sortie are sent to the maintenance routine to calculate either a refuel/rearm time (AIRCRAFT_TYPE(I).REFUEL_REARM_TIME) or a time to repair. Both normal malfunctions and damage caused within the engagement module will be considered. Maintenance will use a random selection process based on aircraft type maintenance information. Until extensive aircraft maintenance data are gathered and accurate curves are fit to those data, Maintenance will use exponential functions to calculate time to repair:

$$\text{Time to Repair} = ae^{rt}$$
$$(\text{TR} = ae^{rt})$$

In this formula, "a" equals ACPARM(I).MAINTIME_DIST(2), and "t" equals ACPARM(I).MAINTIME_DIST(1). Both are variables dependent on the type of aircraft being considered. "R" is a random number generated for each aircraft being considered with a maintenance time less than or equal to two hours (120 minutes). The aircraft will be considered to be repaired during refuel/rearm periods.

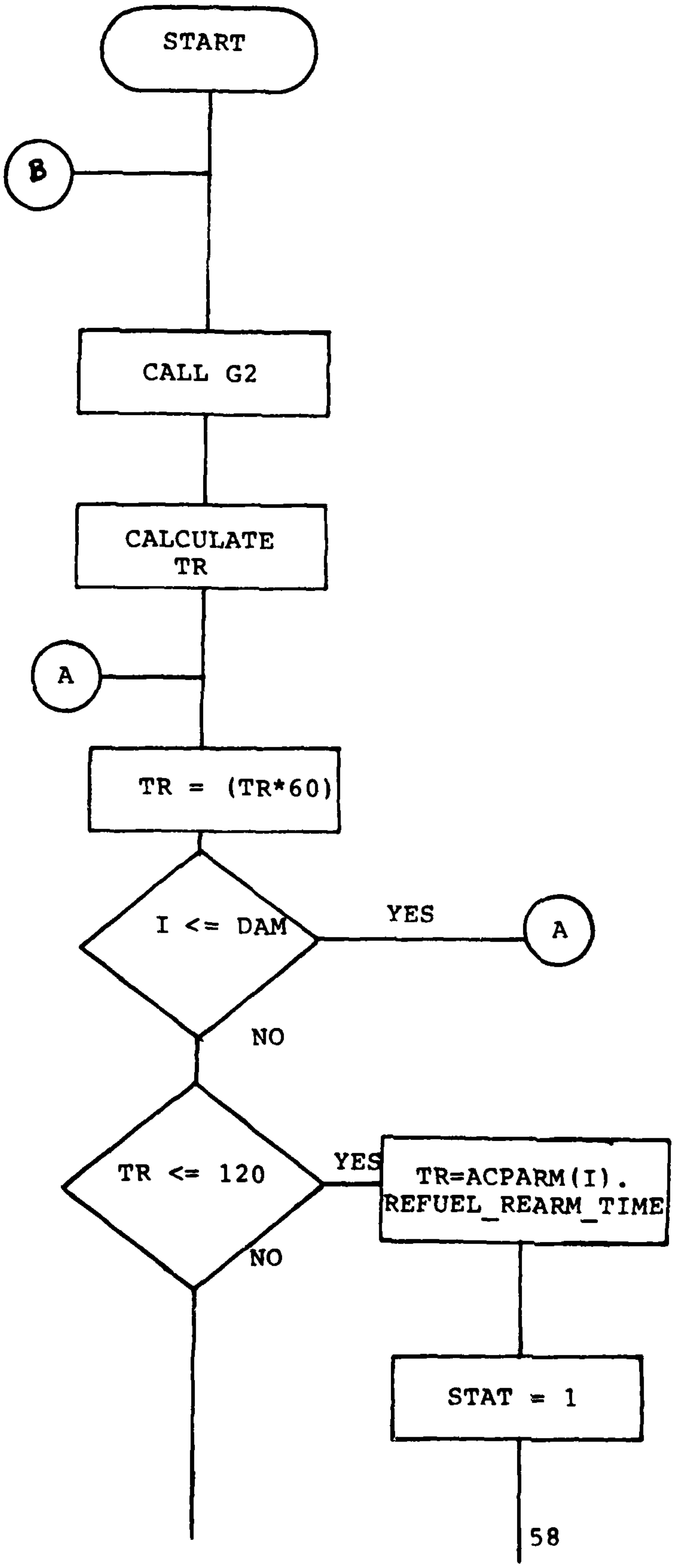
Figure 7 shows the graph of TR with a = 0.1 and t = 6. Using these values, 35% of the aircraft require maintenance time of less than or equal to two hours. A maximum value of just over forty hours is shown. We feel this is satisfactory until more complete data can be obtained.



$TR = .1e^{6R}$

FIGURE 7

4.7.2 CASMNT FLOWCHART



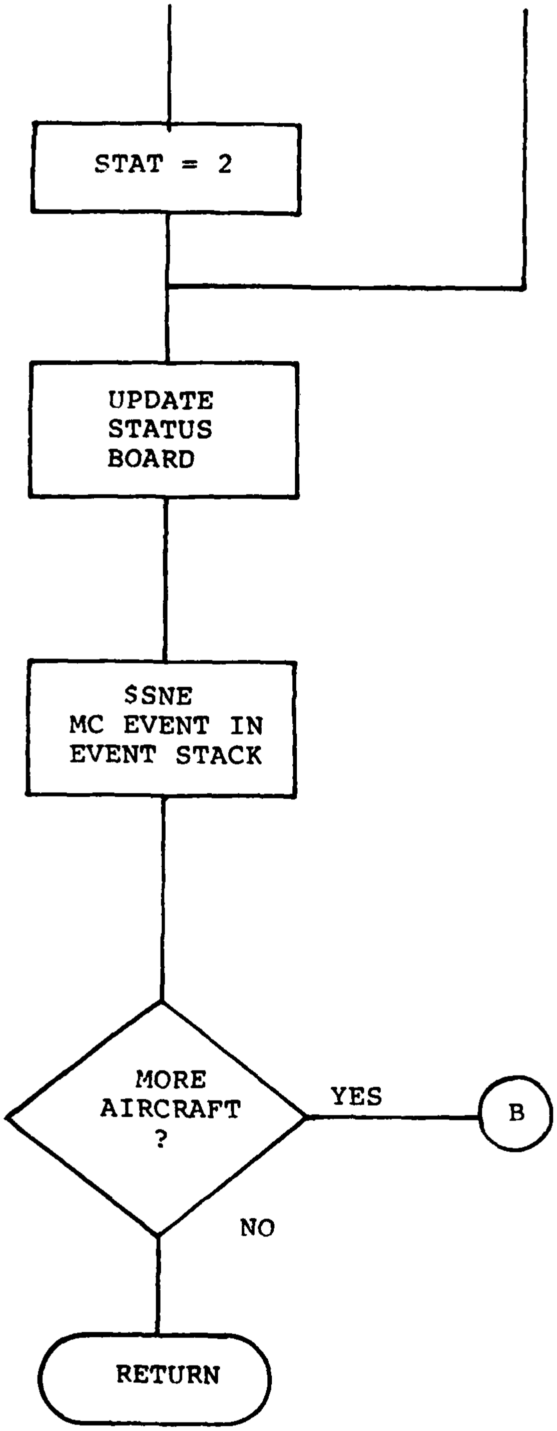
MAINTENANCE called by Launch Routine or Recovery Complete Routine with a list of number and type aircraft for maintenance calculations, e.g.,

Number (NUM)	Type	Damaged (DAM)
4	F14	0
3	A6	1
2	A7	0

$TR = ae^{(R)t}$
R is a random number between 0 and 1 selected for each aircraft.

Calculate TR for damaged aircraft (I = 1 to DAM).

If TR is less than or equal to 2 hours then set the status for the refuel/rearm of the aircraft and set TR to equal the time to refuel and rearm an aircraft; else set the status for the maintenance of aircraft.



Update the turnaround status in the Status Board

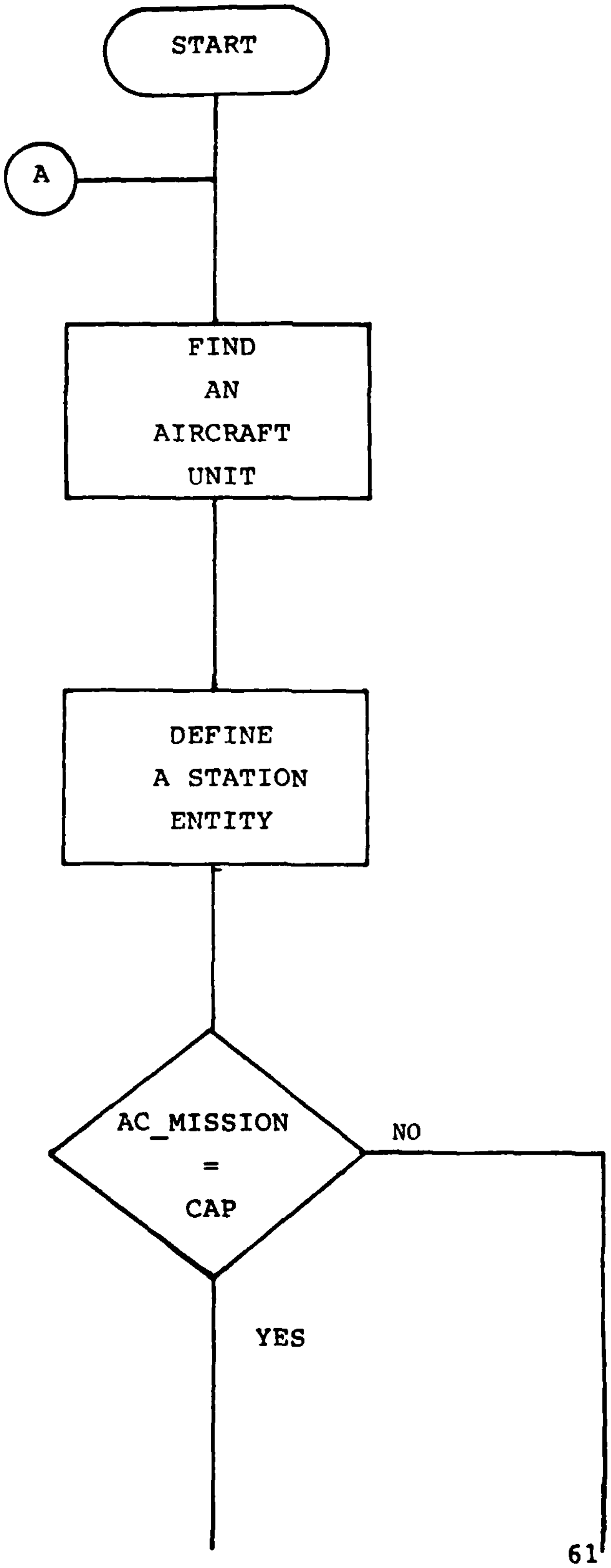
Schedule Maintenance Complete Event at CT + TR.

4.8 Station Routine

4.8.1 Description

The Station Routine, is called by the Launch Complete Event to create the battle group station entities (BG_STATION) for aircraft whose missions are equivalent to CAP, AEW, or VS. The station information for each station aircraft is selected from either one of two 2-dimensional or one 3-dimensional disposition selection table (DISP_CAP, DISP_AEW, or DISP_VS). The data elements needed to select the location in a disposition selection table are the table number, TABLE# which is either '1' or '2' and the number of towed arrays, TOWED_ARRAYS, for VS aircraft. By calling the Create routine and the $LINK macro, the stations are created and linked to the battle group station list (BGROUP.DISPOSITION.BG_STATIONLIST) in order of priority.

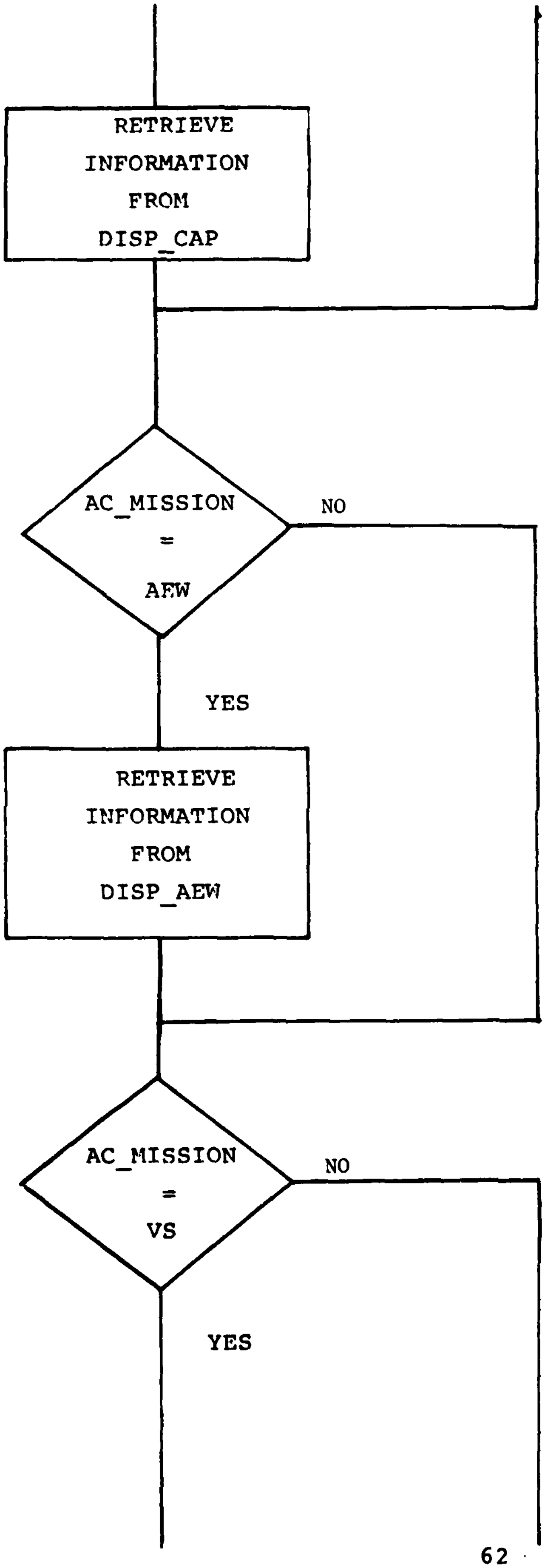
4.8.2 CASSTAT FLOWCHART

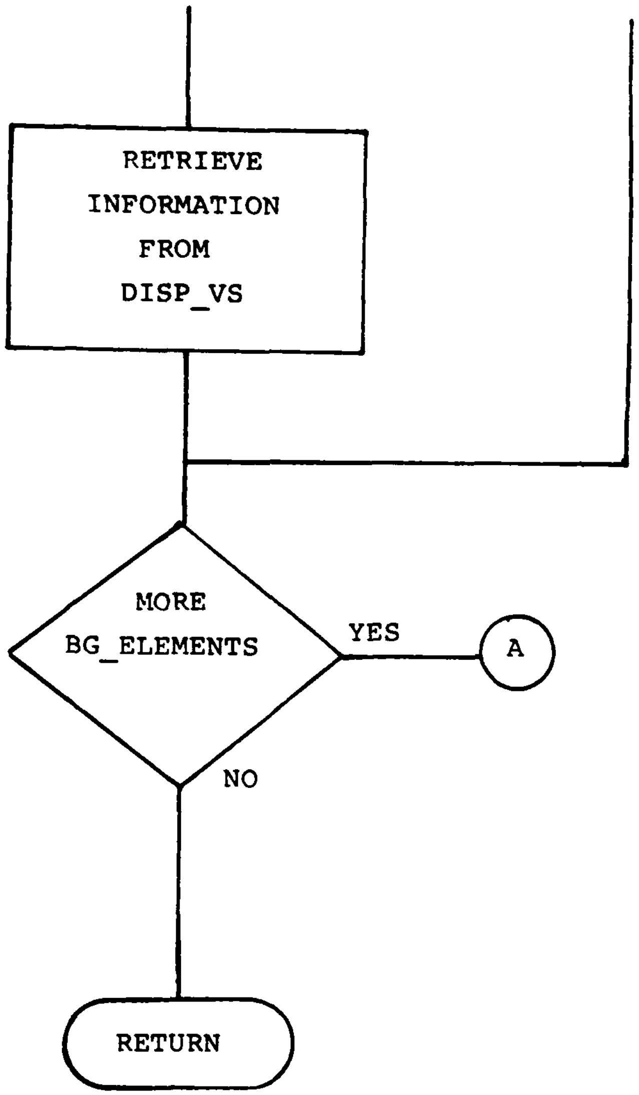


Cvcle through the battle group element list (BGROUP. COMPOSITION)

Find a BG_ELEMENT in the composition list which contains a pointer to an aircraft unit

Using the Create subroutine and the $LINK macro, create a battle group station entity (BG_STATION) and link it to the battle group station list (BGROUP. DISPOSITION.BG_STATIONLIST)


RETRIEVE INFORMATION FROM DISP_CAP
AC_MISSION = AEW
NO
YES
RETRIEVE INFORMATION FROM DISP_AEW
AC_MISSION = VS
NO
YES



Get the next BG_ELEMENT from the battle group composition list

5.0 DATA STRUCTURE RELATIONSHIP

The CASAM Module is designed to use data structures which are compatible with the NNWS structures. Within the data structures there are static and dynamic data types. The static data structures contain data that is held constant for each aircraft. The interrelationships and relationships of these structures is shown in Figure 8.

The aircraft parm (ACPARM) declaration which is initialized under the air raid operations, is used to contain data for calculations in the Launch Routine, the Launch Complete and Recovery Routine, and the Maintenance Routine. This structure is aircraft type dependent and will contain information on each aircraft type referenced by an integer code. The data in this structure will include the aircraft designation (i.e., F14), the missions to which it can be assigned, the ordnance loads it can carry, the abort rate, the maintenance coefficients, the launch rate, the recovery rate, the rearm/refuel time, and speed data. The data structure appears as follows:

```
DCL #ACTYPE              FIXED BINARY(15),
    1 ACPARM (1: #ACTYPE) CONTROLLED,
      2 ACNAME           CHARACTER(8) VARYING,
      2 COLOR            FIXED BINARY(15),
      2 ABORT_RATE       FLOAT BINARY(21),
      2 MAINTIME_DIST(2) FLOAT BINARY(21),
      2 LAUNCH_RATE      FLOAT BINARY(21),
      2 RECOVERY_RATE    FLOAT BINARY(21),
      2 REFUEL_REARM_TIME_DIST FIXED BINARY(31),
      2 ALLOWABLE_MISSION(5)   FIXED BINARY(15),
      2 ALLOWABLE_ORDNANCE(5)  FIXED BINARY(15),
```

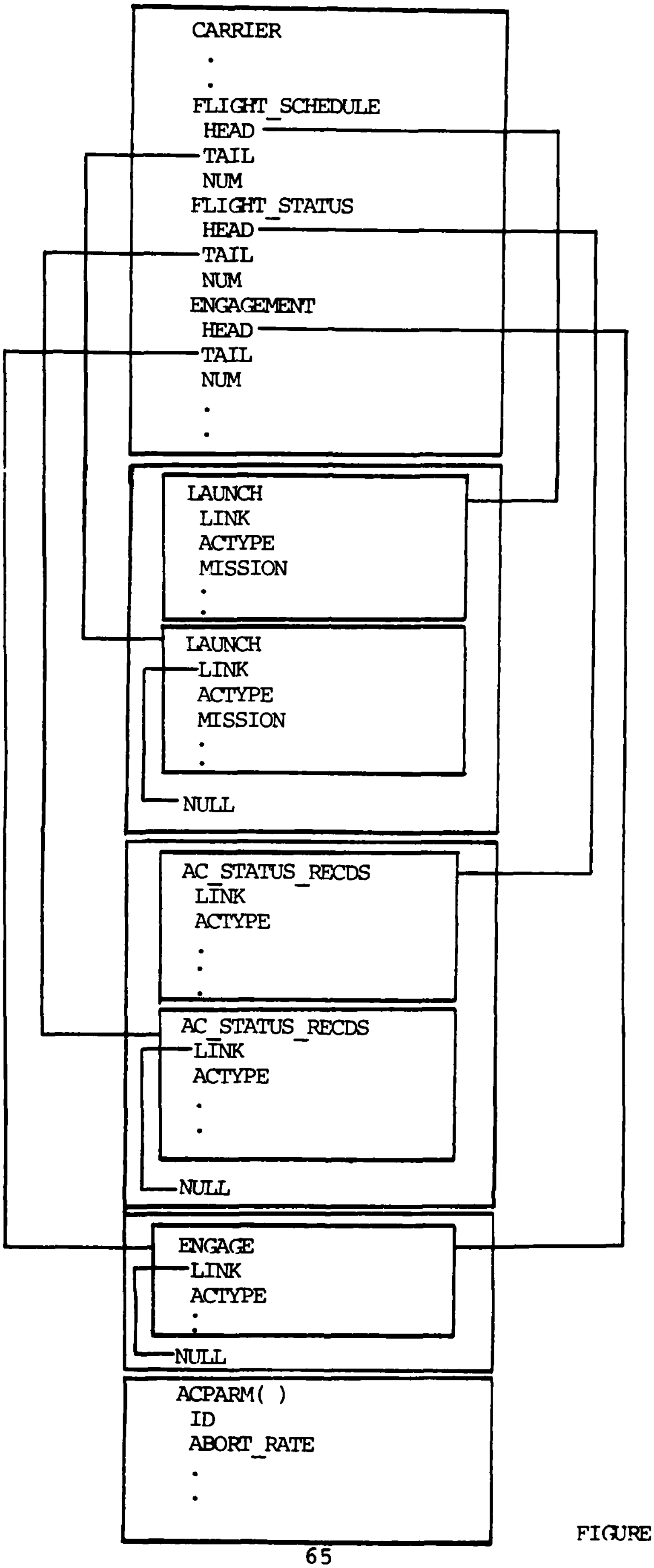

CARRIER
.
.
FLIGHT_SCHEDULE
HEAD
TAIL
NUM
FLIGHT_STATUS
HEAD
TAIL
NUM
ENGAGEMENT
HEAD
TAIL
NUM
.
.
LAUNCH
LINK
ACTYPE
MISSION
.
.
LAUNCH
LINK
ACTYPE
MISSION
.
.
NULL
AC_STATUS_RECDS
LINK
ACTYPE
.
.
.
AC_STATUS_RECDS
LINK
ACTYPE
.
.
NULL
ENGAGE
LINK
ACTYPE
.
.
NULL
ACPARM( )
ID
ABORT_RATE
.
.

FIGURE 8

```
2 ENDURANCE          FIXED BINARY(31),
2 SOA,
  3 CRUISE           FLOAT BINARY(21),
  3 ATTACK           FLOAT BINARY(21).
```

The carrier data structure will contain pointers to three daisey linked lists. These lists are as follows:

- The CARRIER.FLIGHT_SCHEDULE points to a list of launch sequences (the LAUNCH declaration) in which each launch is based on launch time, aircraft type, number of aircraft, mission, and RTB time.

- The CARRIER.STATUS_BOARD points to a list based on aircraft type, which contains the aircraft status on possible missions and on stations (the AC_STATUS_RECDS declaration).

- The CARRIER.ENGAGEMENT points to a list of those aircraft on engagement missions such as strike or attack (the ENGAGE declaration).

The portion of the carrier data structure which needs to be updated appears as follows:

```
DCL  P@CARRIER                     POINTER,
     1 CARRIER                     BASED(P@CARRIER),
         .
         .
         .
     2 SKED#                       FIXED BINARY(15),
     2 AWC                         FIXED BINARY(15),
     2 FLIGHT_SCHEDULE,
       3 HEAD                      OFFSET(TESPACE),
       3 TAIL                      OFFSET(TESPACE),
       3 NUM                       FIXED BINARY(15),
     2 FLIGHT_STATUS,
       3 HEAD                      OFFSET(TESPACE),
       3 TAIL                      OFFSET(TESPACE),
       3 NUM                       FIXED BINARY(15),
     2 ENGAGEMENT
       3 HEAD                      OFFSET(TESPACE),
       3 TAIL                      OFFSET(TESPACE),
       3 NUM                       FIXED BINARY(15),
         .
         .
         .
```

To indicate whether the Launch Complete Event or the Recover Complete Event is being executed, the battlegroup declaration must contain the following data elements:

```
DCL  P@BGROUP                          POINTER,
     1 BGROUP                          BASED(P@BGROUP),
         .
         .
         .
       2 LNCHCMP_RCVRCMP_TIME          FIXED BINARY(31),
         .
         .
         .
```

VARIABLE LIST

AC	maximum number of VF to be assigned to this controller
ACA	number of aircraft available for launch
ACID	blue unit ID of air controller for this CAP station
ACL	number of aircraft launched
ACTYPE	type of aircraft
ADR	automatic decision rules
AL	number of aircraft aborted
ALTD	altitude of blue unit or CAP station
ARL	abort rate for a specific aircraft type
CT	current game time
DC	DEFCON
DAM	number of damaged aircraft of a given type
EAR	engagement aircraft returning
EF	engagement factor
K	number of types of aircraft returning
LOAD	loadtime for aircraft ordnance and fuel
LCE	launch complete event
LR	launch rate for a specific aircraft type
LT	launch time
NUM	number of aircraft of a given type recovered
PT	preparation time in minutes
RTB	return to base time

VARIABLF LIST continued

SAR	station aircraft returning
SKEDNEW	current schedule number
SKEDOLD	previous schedule number
SKED#	total number of schedules
SLIDE TIME	number of minutes to delay launch event
STAT	status of aircraft
TR	time to repair
TTR	total time to recover aircraft
TTSAR	total time for station aircraft returning
TTEAR	total time for engagement aircraft returning
VALID#	schedule number obtained in BGDISP routine which is passed to the Create Schedule Routine
ZULU	Greenwich Mean Time

ACRONYM LIST

AAW	Anti-Air Warfare
A/C	Aircraft
AEW	Airborne Early Warning
ASUW	Anti-Surface Warfare
ASW	Anti-Submarine Warfare
AWC	Airwing Composition
BG	Battle Group
CAP	Combat Air Patrol
CASAM	Carrier Air Strike and Attack Module
CDRL	Contract Data Requirements List
CVBG	Carrier Battle Group
DEFCON	Defense Condition
DLI	Deck Launch Interceptor
EW	Electronic Warfare
GMT	Greenwich Mean Time
K	1,000
MK	Mark
NNWS	Naval Nuclear Warfare Simulation
NADS	Naval Air Defense Simulation
PIM	Position of Intended Movement
SCP	System Change Proposal

REFERENCES

1. Dictionary of Variables for Naval Air Defense Simulation (NADS), TRW, SSP-MCE-OG-0004-81, 30 March 1981.

2. NNWS Preliminary Design Description for Carrier Battle Group and Air Raid Operations (Draft), Applied Physics Laboratory, 30 July 1982.